07-014

Architectural Innovation and Dynamic Competition: The Smaller "Footprint" Strategy

Carliss Y. Baldwin*
Kim B. Clark†

Version 1.0
August 2006

We are grateful to Michael Jacobides for encouraging us to write this paper and to Venkat Kuppuswamy for comments on the draft. We alone are responsible for errors, oversights and faulty reasoning.

*corresponding author:
Harvard Business School,
Boston, MA, 02163
cbaldwin@hbs.edu

† Brigham Young University, Idaho

Architectural Innovation and Dynamic Competition: The Smaller "Footprint" Strategy

by

Carliss Y. Baldwin and Kim B. Clark

Abstract

We describe a dynamic strategy that can be employed by firms capable of architectural innovation. The strategy involves using knowledge of the bottlenecks in an architecture together with the modular operator "splitting" to shrink the "footprint" of the firm's inhouse activities. Modules *not* in the footprint are outsourced—module boundaries are redrawn and interfaces designed for this purpose. The result is an invested capital advantage, which can be used to drive the returns of competitors below their cost of capital. We explain how this strategy works and model its impact on competition through successive stages of industry evolution. We then show how this strategy was used by Sun Microsystems against Apollo Computer in the 1980s and by Dell against Compaq and other personal computer makers in the 1990s.

Key words: architecture — innovation — knowledge — modularity — dynamics — competition — industry evolution

JEL Classification: D23, L22, L23, M11, O31, O34, P13

Architectural Innovation and Dynamic Competition: Why the Smaller Footprint Wins

1. Introduction

In this paper, we propose a model of dynamic competition based on one firm's investment in architectural innovation. By studying the underlying cause-and-effect relationships in a complex architecture, a firm can identify "bottlenecks" and redesign the interfaces of key components to make them more modular. With knowledge of bottlenecks and potential new modules, the firm can then "shrink its footprint," that is, it can outsource more activities without sacrificing either performance or cost. As a result, the firm in question can offer competitive products (or services) but invest less: it will enjoy an "invested capital advantage" over its competitors.

The paper is divided into three parts. In the first (sections 2 and 3), we explain how architectural knowledge combined with use of the modular operator "splitting" (Baldwin and Clark, 2000) allows a firm to reduce invested capital with no degradation in the performance or cost of its products. We relate this idea to a design principle in computer architecture known as Amdahl's Law.

In the second part (sections 4-6), we set up a simple, formal model of industry pricing and dynamics and use it to explore how one firm's invested capital advantage affects dynamic competition. Initially, we show, all entrants will grow, but the one with the invested capital advantage will grow faster. Over time, prices will decline, but for awhile all firms will enjoy apparently healthy rates of growth and profitability. The game changes when the price falls to the point where the weaker firms can no longer earn their cost of capital. At this point (subject to caveats we will describe), the firm with the invested capital advantage will keep growing, while its rivals will begin to disinvest and shrink. (In extensions of the basic model, we discuss what happens if the rivals keep investing.) Under the model's simplified assumptions, the end result is a product market dominated by a single firm.

In the third part of the paper (sections 7-9), we describe two cases that we believe fit the pattern of "footprint competition" based on superior architectural knowledge. The first case involves

Sun Microsystems and Apollo Computer in the late 1980s. Here we have information on how Sun used architectural knowledge to achieve a smaller footprint, and thus we can tie Sun's invested capital advantage to specific architectural innovations. The second case involves the competition between Dell and Compaq (and other PC makers) in the late 1990s. In this case, we have indirect evidence that architectural knowledge and innovation contributed in important ways to Dell's success.

2. The Knowledge Behind Architectural Innovation

The concept of architectural innovation was first proposed in the management literature by Henderson and Clark (1990). They define such innovations as follows:

> [Architectural] innovations … change the way in which the components of a product are linked together, while leaving the core design concepts (and thus the basic knowledge underlying the components) untouched (p. 10).

In other words, architectural innovation involves rearranging known parts (components) into new patterns (architectures) to achieve higher levels of system performance on one or more dimensions.

Henderson and Clark go on to claim that:

> Architectural innovation destroys the usefulness of [the non-innovator's] architectural knowledge, but preserves the usefulness of its knowledge about the product's components (p. 10).

Implicitly architectural innovations depend on the innovator's superior architectural knowledge. But what *is* architectural knowledge? What do those with superior architectural knowledge know?

Let us back up and ask: what is an architecture? Ulrich (1995, p. 419) defines a product architecture as "the scheme by which the function of a product is allocated to physical components." Thus architectural knowledge encompasses knowledge about *the functions of the system* and *how the components of the system contribute to those functions*. (Although Ulrich was talking specifically about product architectures, his definition can be applied to any complex system.) Baldwin and Clark (2000, p. 77) similarly define a system's architecture as the "modules that will be part of the system, and what their roles will be." They treat the interfaces of a system, that is, the detailed descriptions of how

the modules will interact, as separate from the architecture, but part of the system's design rules.

In contrast, Fixson (2005) includes both the function-to-component mapping *and* the system's interfaces in his "operational definition" of an architecture. And, importantly, MIT's Engineering Systems Department (ESD) Committee on Architecture (2004) defines architecture as: "an abstract description of the entitites of a system and how they are related" (p. 2). They go on to define several types of architectures (p. 5):

- The *functional* architecture (a partially ordered list of activities or functions that are needed to accomplish the system's requirements);
- The *physical* architecture (at minimum a node-arc representation of physical resources and their interconnections);
- The *technical* architecture (an elaboration of the physical architecture that comprises a minimal set of rules governing the arrangement, interconnections, and interdependence of the elements, such that the system will achieve the requirements);
- The *dynamic operational* architecture (a description of how the elements operate and interact over time while achieving the goals).

Note that Ulrich's function-to-component mapping is subsumed in the functional and dynamic operational architectures, while interfaces are subsumed in the physical and technical architectures.

Following the ESD Committee on Architecture's view, we define *architectural knowledge* to be knowledge about "the entities of a system and how they are related." On this view, architectural knowledge comprises knowledge of (1) how the system performs its functions (the function-to-component mapping); (2) how the components are linked together (the interfaces between components); and (3) the behavior of the system, both planned and unplanned, in different environments. Gaining superior architectural knowledge involves mapping the system's functions onto its elements and their patterns of linkage, and investigating how changes in the elements or patterns of linkage affect performance on various dimensions. To this end, architects can experiment with putting the system together in different ways, in each case measuring how its performance changes. They can also study the system under different conditions and meter its internal operation to see what levels of activity or stress arise at different junctures.

System designers can learn three important things from investments in architectural

knowledge. First, they may discover that some dimension of performance is constrained by a bottleneck involving one or more components. Second, they may find that one or more components, arranged in a new pattern, will deliver higher levels of performance. Last but not least, they may learn how to separate components from the rest of the system and encapsulate them behind well-specified interfaces. Components separated in this way become *modules* of the system (Parnas, 1972; Parnas et.al., 1985; Baldwin and Clark, 2000).

Knowledge of bottlenecks, as we shall see, is crucial to achieving architectural innovation. In complex systems, there are two types of bottlenecks. We label them *absolute bottlenecks* and *fractional bottlenecks* for reasons that will become clear below.

2.1 Absolute bottlenecks.

The performance of a complex system (on some dimension) may equal the performance of its least-good component. Symbolically, let X denote the performance the system and $x_1, ..., x_n$ denote the performances of each of n separate components. An absolute bottleneck exists if:

$$X = \min(x_1, ..., x_n) \quad . \tag{1}$$

Architectures subject to absolute bottlenecks are common in the real world. For example, an assembly line is only as fast as its slowest station; the security of a system is only as good as its most vulnerable portal; a chain is only as strong as its weakest link.

In a system with an absolute bottleneck, there is no point in seeking to improve any part of the system that does not involve the bottleneck. Remedying the bottleneck, however, may involve more components than those that are directly responsible for it. For example, if the bottleneck is a capacity constraint, the architects can sometimes shift the capacity burden to other parts of the system. (In process engineering, this is known as "line-balancing.") Or the architects can directly add capacity at the bottleneck itself, for example, by putting two components where there was one.

If rebalancing and adding capacity at the bottleneck are not feasible, then the only way to address the bottleneck is to redesign its elements. In this case, architectural knowledge (of the location of the bottleneck) is needed to identify the problem, *but component knowledge is needed to devise*

a solution. And if a solution is found, it will appear to be a "modular innovation" in the Henderson and Clark classification scheme. But the fact that the innovation addresses a bottleneck makes it an architectural innovation as well.

In a system with absolute bottlenecks, opportunities for innovation will shift rather abruptly: as one bottleneck is relaxed, another will emerge. Again, architectural knowledge is needed to know where the new bottleneck is, but component knowledge may be needed to devise a remedy.

2.2 Fractional bottlenecks and Amdahl's Law

Instead of "weakest link" performance and an absolute bottleneck, some systems exhibit *additive performance*. In this case, system performance equals the sum of the performances of individual components. Using the same notation as above:

$$X = x_1 + ... + x_n \qquad (2)$$

Architectures with additive performance are also common in the real world. For example, the processing time for an instruction in a computer equals the sum of the times the instruction spends in various states. The time needed to run a software program is the sum of the times needed to complete its instructions.[1] And the cost of making a product equals the sum of the costs of each input.

In systems with additive performance, although all components *contribute* to the whole, they are not all equally interesting targets of innovation. In computer architecture, there is a maxim known as Amdahl's Law, which can be succinctly stated as "make the common case fast."[2] Although it derives from computer science, Amdahl's Law is in fact a general principle that can be applied to any system with additive performance.

To see how Amdahl's guides architectural innovation,[3] divide equation (2) by total system

---

[1] In general, architectures based on parallel processing are subject to absolute bottlenecks, while those based on sequential processing are subject to fractional bottlenecks.

[2] The technical statement of Amdahl's Law is: "The speedup to the entire system caused by the enhancement [of some part] is limited by the fraction of computation time in the original machine that can be converted to take advantage of the enhancement." The phrase "make the common case fast" is due to Hennessy and Patterson (1990).

[3] This example is based on Hennessy and Patterson (1990), pp. 8-12.

performance, X. The result shows overall performance (normalized to one) expressed as a sum of fractions, $F_1, ..., F_n$:

$$1 = F_1 + ... + F_n \quad .$$

Suppose the performance we care about is speed, and a particular innovation will increase the speed of the first component by a factor $s_1 > 1$. (In computer architecture, s_1 is called the "speedup" of component 1.) The speedup of the entire system, S, as a result of speeding up component 1 is:

$$S = \frac{1}{\frac{F_1}{s_1} + ... + F_n} = \frac{1}{\frac{F_1}{s_1} + (1 - F_1)} \quad .$$

Now consider the interaction of s_1 and F_1. If the fraction of time accounted for by the component is large then most of the component's speedup will be translated to the system level. But if the fraction of time accounted for by the component is small, then the component's speedup will have very little effect on system performance. For example, suppose $s_1 = 2$. If $F_1 = 90\%$, then $S = 1.82$, and system performance will go up by 82%. But if $F_1 = 1\%$, then $S = 1.005$, and system performance will go up by only .5%. That is to say, a large increase in the speed of a small component has almost no effect on the performance of the system as a whole.

Components that account for a large fraction of additive performance are *fractional bottlenecks*. As with absolute bottlenecks, to remedy a fractional bottleneck, architects can link the existing components in new ways or add new components at the bottleneck. But if these remedies fail, then it will be necessary to redesign the components themselves. Again, architectural knowledge (of the fractional weights) is necessary to identify the problem, but component knowledge may be needed to solve it. And the resulting innovation will look like a modular innovation in the Henderson and Clark classification scheme, even though it rests in part on architectural knowledge.

Fractional bottlenecks are not as stark as absolute bottlenecks. Improving any component will improve system performance to some degree: the issue is simply which components carry the "most bang for the buck." And as one component is improved, its fractional share will decline, and other fractional bottlenecks will become more important targets of innovation.

2.3 Modularization

Investments in architectural knowledge also allow designers to change the modular structure of the system. A complex system can be envisioned as a set of elements connected by dependencies or links (Steward, 1981; Eppinger, 1991; Pahl and Beitz, 1996; Baldwin and Clark, 2000, 2006) The dependencies can be physical, energetic, or informational: some will contribute to the overall performance of the system; others will detract from or threaten system performance. For example, in a computer, the CPU is connected to other parts of the system via specific pin connectors. The pins create physical, energetic and informational linkages that are crucial to the functioning of the whole. But the CPU also generates heat, an unwanted byproduct that can threaten the functioning of the whole. Thus cooling systems are a necessary component of a computer's architecture.

The dependencies inherent in the architecture of a complex system must be mirrored in the process of designing the system (Baldwin and Clark, 2000). If component 1 is physically next to, or supplies energy to, or provides information to component 2, the designers of the two components must take that fact into account.

Such dependencies can be managed in two different ways, however. The first is for the designers to communicate in real time and work out by mutual adjustment how to handle the dependencies. The second is for an architect to specify *ex ante* what dependencies will exist. (The specification can be in terms of bounds and tolerances). This specification transforms a negotiated set of dependencies into a rule—a *design rule*—that is binding on both sets of designers. In the presence of a mutually binding rule, the designers do not have to communicate with one another (Baldwin and Clark, 2000).

Design rules place more restrictions on the final architecture of the system than dependencies based on real-time communication and mutual adjustment. There is, by definition, no room in a rule for give-and-take, and thus some new ideas may not be accommodated. Notwithstanding this fact, putting design rules in place does not necessarily hurt system performance. Architectural knowledge can be used to determine *which* design rules offer little or no harm to the overall system.

In the first place, architectural knowledge includes knowledge of the performance of prior designs (Bell and Newell, 1971, p. 87; Baldwin and Clark, 2000). From this knowledge, architects may know that a particular configuration works well enough, that is, clears some performance threshold. There is then no need for the designers to reinvent the wheel: the configuration in question can be specified in advance as a design rule. Such a rule saves the designers time and effort: it makes the design process more efficient without detracting from the quality of the system.

Knowledge of bottlenecks—another form of architectural knowledge—also helps in the placement of design rules. For example, suppose architectural knowledge reveals that (1) the system has an absolute bottleneck, but (2) component 1 is *not* part of it. The architects can then create design rules to isolate component 1 from the rest of the system. This is the modular operator called "splitting" (Baldwin and Clark, 2000). The result of splitting is a *module* that is encapsulated behind its interface. Key information about the module is *hidden* from the designers of the rest of the system (Parnas, 1972).

Component 1's functionality in the system may be harmed by this modularization. *But as long as this degradation is not too severe, system performance will not suffer because component 1 was not in the bottleneck*. The same principle applies to fractional bottlenecks. Architects can split off components with low fractional contributions and make them modules, with little or no damage to the performance of the system as a whole.

3. The Strategic Use of Architectural Knowledge

Suppose Firm *A* has invested in architectural knowledge about its product and production process. As a result of this investment, its designers know about the system's bottlenecks and have ideas about how to remedy them. They also know how to convert some non-bottleneck components into modules.

We assume that Firm *A*'s competitors have not made a comparable investment, thus do not have such knowledge. Such asymmetries in architectural knowledge among competitors are

common, hence we think this assumption is appropriate. Gaining architectural knowledge is expensive, and its benefits are not always obvious to non-technical managers or financiers. At the same time architects are often steeped in technical knowledge but do not know how to convert their knowledge into an effective corporate strategy. Indeed one of the goals of this paper is to clarify the connections between architectural knowledge and strategy so that the "business case" for investing in such knowledge can be strengthened.

What should Firm *A* do with its superior architectural knowledge? One option would be to design and build a better system with no commensurate increase in costs. Firm *A* would then be in a position to offer a quality-differentiated product on the market. Shaked and Sutton (1983) and Sutton (1991) model quality- differentiated competition based on sunk cost investments. They argue that the end result is likely to be an oligopoly consisting of one or two dominant firms and a fringe of competitors.

Alternatively Firm *A* might combine its knowledge of bottlenecks and (potential) modules to create a *more modular system* with equivalent performance and cost. This can be achieved by (1) implementing a superior design at the bottleneck; (2) and creating standardized interfaces between the system and non-bottleneck components. (The designers may give up some performance in the non-bottleneck components in return for modularity.) A system with more modules perforce has more "thin crossing points," that is, places in the product design and/or production process where the dependencies between components are few and simple.

The thin crossing points in a modular architecture amount to "technologically separable interfaces" in Williamson's (1985) theory of transaction costs. Thin crossing points have low "mundane" transaction costs,[4] and thus, *ceteris paribus*, are good locations for transactions (Baldwin and Clark, 2006; Langlois, 2006). Hence, under this option, Firm *A* can place transactions at the thin crossing points of its modular architecture and *outsource modules that are not in the bottleneck*.

[4] "Mundane" transaction costs are the costs of standardizing, counting, valuing and paying for whatever crosses a boundary or interface. These actions are required if a transaction is to take place. (Baldwin and Clark, 2006.)

The modularity-cum-outsourcing strategy requires both a new system design and new contracts with suppliers. Thus it involves both design costs and transaction costs. But, with its superior architectural knowledge, Firm A can *choose* what to split off and what to keep inhouse. Thus, in the presence of superior architectural knowledge, modularization and outsourcing may be accomplished without decreasing the performance of the system or increasing its cost. The end result will be a system whose performance and cost equal that of the competition, but with a smaller *footprint*.

We define the *footprint* of a product as the set of activities performed by the firm that sells it. It is what the seller decides to insource. The *invested capital* of a particular footprint equals the capital needed to support these insourced activities. It includes both working capital—receivables and inventory minus payables—and property, plant and equipment. In general, larger footprints, that is, more insourced activities, require more invested capital.

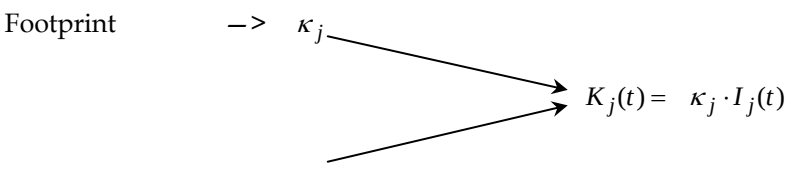
In what follows, we will need to track the invested capital of a firm over time, as well as its capacity (for sales) in aggregate and per unit of invested capital. Thus we introduce the following notation:

$I_j(t) \equiv$ Firm *j*'s invested capital at time *t*;

$K_j(t) \equiv$ Firm *j*'s capacity at time *t* (the maximum number of units it can produce and sell in the interval between *t* and *t+1*);

$\kappa_j = \frac{K_j(t)}{I_j(t)} \equiv$ Firm *j*'s capacity-to-invested-capital ratio (in practice, this is called the "asset turnover ratio").

The firm's choice of a footprint determines its asset turnover ratio κ_j. Its past investment decisions determine its invested capital at time *t*. And its turnover (κ_j) ratio times its invested capital equals its capacity:



Prior Investment $\rightarrow I_j(t)$

By definition, a *small footprint* strategy, based on superior architectural knowledge, increases the firm's asset turnover ratio. Firm *j* can then produce more units with less capital than its competitors: it has an *invested capital advantage*. In the next three sections, we use a simple model to investigate what one firm's invested capital advantage implies for the structure and evolution of its industry in the short and long run.

4. A Model of Industry Dynamics When One Firm Has an Invested Capital Advantage

In this section we describe the base case assumptions of our formal model of dynamic competition.

4.1 Products and Technology

Assume that two firms, $j = (A, B)$, make and sell a product. (The two-firm assumption is made for expositional convenience and is not essential.) The quality of the product and unit costs of production are the same for both firms. However, because of a prior investment in architectural knowledge combined with selective outsourcing, *A* has a smaller footprint. Thus *A*'s asset turnover ratio is higher than *B*'s:

$$\kappa_A > \kappa_B \quad .$$

We assume this ratio is constant through time.

Both firms' invested capital depreciates at the rate δ per period. If depreciated capital is not replaced, the firm's capacity will shrink:

$$K_j(t+1) = \quad (1-\delta) \cdot K_j(t) \quad .$$

4.2 Demand

Demand for the product is characterized by a downward-sloping demand function, $Q(p)$ where $Q'(p) < 0$. All customers are aware of both firms, and the firms compete on the basis of price. The firms do not price-discriminate: in any time interval all customers buying goods from a

given firm pay the same price.

4.3 Financing

The firms start off with the same amount of equity capital. However, because of its higher turnover (higher κ), *A* has more initial capacity than *B*:

$$I_A(0) = I_B(0) \quad \Rightarrow \quad \kappa_A I_A(0) > \kappa_B I_B(0) \quad \Rightarrow \quad K_A(0) > K_B(0) \quad .$$

For simplicity, we assume that the two firms do not issue debt, do not pay dividends, and do not receive further infusions of equity after their founding. (These assumptions will be relaxed below.)

4.4 Timing

Time is marked out in discrete intervals, $(t, t+1)$. At the start of each interval, each firm has a certain amount of invested capital, $I_j(t)$, which determines its capacity, $K_j(t)$. The firms set prices at the start of the interval, and then receive orders, which they fulfill during the interval. Cash (from the collection of receivables, which are included in the invested capital base) comes in during the interval. Each firms can use its cash receipts to replace depreciated capital and add to its invested capital base. (Consistent with standard accounting practice, depreciation expense is deducted from earnings, hence is part of the unit cost of the product.)

At the beginning of the next interval, each firm's capacity will equal will equal its old capacity, $K_j(t)$, less depreciated capacity, $\delta K_j(t)$, plus replacement capacity, $R_j(t)$, plus new capacity purchased with its incremental investment, $\kappa_j \Delta I_j(t)$:

$$K_j(t+1) = K_j(t) - \delta K_j(t) + [R_j(t) + \kappa_j \Delta I_j(t)] \quad .$$

The term in brackets indicates capital expenditures that are at the discretion of the firm.

As long as the firm elects to grow, replacement capital will equal depreciated capital and thus:

$$K_j(t+1) = K_j(t) + \kappa_j \Delta I_j(t) \quad .$$

But if the firm chooses not to invest any more in the business, its capacity will shrink:

$$K_j(t+1) = K_j(t) - \delta K_j(t) \quad .$$

Non-invested earnings and depreciation cash flow accumulate in a cash account.

Finally, under the assumption that no new debt or equity is issued, $\Delta I_j(t)$ must be less than or equal to *j*'s after-tax earnings from the period $(t, t+1)$:

$$\Delta I_j(t) \quad \leq \quad \text{Net Income} \quad \equiv \quad (1-\tau)\cdot[p_j(t)-c]\cdot Q_j(t) \quad ;$$

where:

- $p_j(t)$ is the price charged by Firm *j* during the interval *(t,t+1)*;
- c is the unit cost of the product (common to both firms);
- $Q_j(t)$ is the number of unit sold by Firm *j* during the interval; and
- τ is the tax rate on corporate profits.

4.5 Return on Invested Capital (ROIC).

For the time interval $(t, t+1)$, we define Firm *j's* return on invested capital as:

$$ROIC_j(t) \quad \equiv \quad \frac{\text{Net Income}}{\text{Invested Capital}} \quad \equiv \quad \frac{(1-\tau)\cdot[p_j(t)-c]\cdot Q_j(t)}{I_j(t)} \quad .$$

Here the numerator is Firm *j*'s after tax profit during the interval and the denominator is the the invested capital needed to obtain these profits.

A firm's *ROIC* can be compared to the cost of capital for assets of comparable risk. If a firm's *ROIC* is greater than (or equal to) the cost of capital, then the firm is an attractive opportunity for investors. If its *ROIC* is less than the cost of capital, investors would be better off purchasing other assets of equivalent risk in the capital markets. (Note: if a firm issues no debt, as we have assumed, its *ROIC* equals its return on equity, and the comparison to market rates of return is simple. If the firm issues debt, then it is necessary to take debt tax shields into account. Our model abstracts from this complexity.)

4.6 Decision Rules

In each period, the firms must decide on prices and on investment. As our base case, we assume that the firms *set prices to utilize all their capacity*. This means that they do not engage in any strategic pricing games.

With respect to investment, we assume the firms are *myopic value maximizers*. By this we

mean that they cannot forecast future prices (although they do know the current price). They do, however, know their cost of capital, and act in accordance with the wishes of investors. Formally, let ρ denote the cost of capital. The firms' investment decision rule is then:

- If $ROIC_j \geq \rho$, Invest, in which case: $K_j(t+1) = K_j(t) + \kappa_j \Delta I_j(t, t+1)$;
- If $ROIC_j < \rho$, Do not Invest, in which case: $K_j(t+1) = K_j(t) - \delta K_j(t)$.

This decision rule will change if the firms behave strategically, or can forecast prices, or are subject to agency problems. Below we will explore how such changes affect the dynamics of industry evolution. But first, as a base case, we consider how competition unfolds under these simple rules.

5. Industry Dynamics: The Base Case

We are now in a position to prove two propositions. The first determines equilibrium industry prices, the second determines each firm's maximum growth rate assuming no external financing.

Proposition 1. We assume that customers buy first from the cheapest supplier. In that case, in the time interval $(t, t+1)$, both firms will charge the same price, which clears the market:

$$p^*(t) \equiv Q^{-1}[K_A(t) + K_B(t)] \quad .$$

Here $K_A(t) + K_B(t)$ equals the sum of A's and B's capacity at the start of the interval, and Q^{-1} denotes the inverse function of Q: $Q^{-1}Q(p) = p$.

Proof. Total units sold, Q , cannot be higher than $K_A(t) + K_B(t)$ because the firms cannot make and sell products beyond their capacity. Thus a firm charging a price below $p^*(t)$ can increase profits without reducing its own unit sales, by raising prices.

If both firms set their prices above $p^*(t)$, then the one with the higher price will have excess capacity. Under the decision rule, it will drop its price until all its capacity is utilized. At that point, the other firm will have excess capacity and will drop its price. The price declines will continue until $p_A(t) = p_B(t) = p^*(t)$. *QED*

For Proposition 1 to hold, (1) customers with the highest willingness to pay must buy from the cheapest firm; and (2) both firms must behave non-strategically. If customers with the highest willingness to pay buy from the more expensive firm, then it is possible for both firms to sell out their capacity while one of them charges a price above $p^*(t)$. If the firms behave strategically, then, for low levels of capacity, there is a Nash equilibrium in which both firms charge $p^*(t)$. For higher levels of capacity, however, there is no equilibrium in pure strategies. Kreps and Scheinkman (1983) have shown that there is a mixed-strategy equilibrium in which both firms name prices above $p^*(t)$ with positive probability, and the larger firm's prices stochastically dominate the smaller firm's. (That is, the larger firm charges higher prices on average, hence has more excess capacity on average than the smaller firm.)

However, the mixed-strategy equilibrium imposes huge common knowledge requirements on the two firms (Samuelson, 2004). They must first know (and know that the other knows) that they are playing the game. They must then jointly determine the distribution functions of their respective equilibrium strategies. We think the probability that two firms engaged in dynamic competition would be able to achieve and sustain this equilibrium is remote, thus we have not made it our base case.

Proposition 2 is a well-known tenet of corporate finance. Let $g_j(t)$ denote the maximum growth rate in the sales of Firm *j* over the interval $(t, t+1)$.

Proposition 2. Assuming no debt and no external equity infusions:

$g_j(t) = ROIC_j(t)$.

Proof. If the firm pays no dividends (as we have assumed), then all of its profit is available for investment in the business:

$$\max \Delta I_j(t) = (1-\tau)\cdot[p_j(t) - c]\cdot K_j(t) \quad .$$

The firm's maximum new capacity at the end of the interval $(t, t+1)$ is thus:

$$\max \Delta K_j(t) = \kappa_j \cdot (1-\tau) \cdot [p_j(t) - c] \cdot K_j(t) \quad .$$

But $\kappa_j = K_j(t)/\, I_j(t)$. Substituting for κ_j, and dividing through by $K_j(t)$, we have:

$$g_j(t) \quad \equiv \quad \max \frac{\Delta K_j(t)}{K_j(t)} \quad = \quad \frac{(1-\tau) \cdot [p_j(t) - c] \cdot K_j(t)}{I_j(t)} \quad \equiv \quad ROIC_j(t) \quad .$$

QED.

5.1 First Epoch of Competition

Propositions 1 and 2 together are sufficient to characterize the industry's dynamics during the first "epoch" of competition, when both firms' *ROIC* s are greater than their cost of capital. During this time:

- Both firms will grow, but *A* will grow faster than *B*, hence *A*'s market share will increase;
- The growth rate of the industry will be a capacity-weighted average of the two firms' *ROIC*s:

$$\text{Industry Growth} \quad = \quad \frac{K_A}{K_A + K_B} \cdot ROIC_A \quad + \quad \frac{K_B}{K_A + K_B} \cdot ROIC_B \quad ;$$

Prices and thus *ROIC*s and growth rates will decline over time. This pattern will continue until *B*'s *ROIC* drops below the cost of capital:

$$ROIC_B(t) \quad = \quad \frac{(1-\tau) \cdot [p^*(t) - c] \cdot K_B(t)}{I_B(t)} \quad < \quad \rho \quad .$$

At this point, the industry enters the second epoch of competition.

5.2 Second Epoch of Competition

Under the assumption of myopic value maximization, when *B*'s *ROIC* drops below the cost of capital, it will begin to disinvest:

$$K_B(t+1) = K_B(t) - \delta K_B(t) \quad .$$

Firm *A* meanwhile will continue to grow at its *ROIC*:

$$K_A(t+1) \quad = \quad K_A(t) + \kappa_A \Delta I_A(t) \quad = \quad K_A(t) \cdot [1 + ROIC_A(t)] \quad .$$

During this transition, aggregate capacity will shrink if:

$$K_A(t) \cdot ROIC_A(t) \quad < \quad \delta_B K_B(t) \quad .$$

This inequality occurs if *A*'s new capacity is insufficient to offset the capacity not being replaced by *B*. If this happens, the market clearing price during the interval $(t+1, t+2)$ will go up. Indeed it may go up enough to make Firm *B*'s *ROIC* higher than its cost of capital, in which case, *B* will begin investing again.

But *A*'s capacity just keeps increasing, until at some point:

$$K_A(t) \cdot ROIC_A(t) \quad \geq \quad \delta_B K_B(t) \quad .$$

Then and thereafter, it will not be profitable for *B* to invest to replace its capacity. This marks the beginning of the third epoch of competition. (The second epoch may not last very long.)

5.3 Third Epoch of Competition

During this period, *A* is growing at its *ROIC* and *B* is shrinking. The total industry growth rate is again a capacity-weighted average of the two firm's growth rates, but *B*'s growth rate is negative:

$$\text{Industry Growth} \quad = \quad \frac{K_A}{K_A + K_B} \cdot ROIC_A \quad + \quad \frac{K_B}{K_A + K_B} \cdot (-\delta) \quad ;$$

Compared with the first and second epochs, industry growth slows down, because *A* must invest to support sales to *B*'s former customers. However, at some point (we assume) *B* will no longer be viable, and will withdraw from the market. This marks the beginning of the fourth and last epoch, when *A* faces no competition.

5.4 Fourth Epoch of (No) Competition

With no competition, *A* is free to behave as a monopolist. It can then set price in one of two ways: First, if there are no other potential entrants, then the market is *not contestable* (Baumol, 1982; Baumol et.al. 1983). *A* can then set its price to maximize monopoly profits:

$$p_A^{**}(t) \quad = \quad c + \frac{Q(t)}{Q'(t)} \quad ;$$

where $Q'(t) \equiv \partial Q(t) / \partial p(t)$. However, if there are potential entrants, then the market is contestable, and *A* must set its price at a level that deters entry. If there are firms with the same architectural

knowledge as *B*, this means setting the price just below the point where their *ROIC* equals the cost of capital:

$$p_A^{**}(t) \quad < \quad p \text{ such that } \quad ROIC_B = \rho \qquad .$$

5.5 Cash Flow Patterns and Financial Returns

As long as the two firms are growing at their respective *ROIC*s, neither can pay dividends to their founding investors. Once a firm stops growing, however, it no longer needs to use earnings to finance growth, and its investors can start to realize cash returns.

How will the investors then fare over the lifecycle of the industry?

Obviously, Firm *A*'s investors will do better than *B*'s, for *A* gains more per unit invested in every period. However, as long as (1) *B*'s managers behave as myopic value maximizers, and (2) *B*'s invested capital can be liquidated without penalty, *B*'s investors may do quite well. The reason is that their initial investment will earn more than the cost of capital for some amount of time (the time it takes the two firms to move down the demand curve). And once the price falls below *B*'s *ROIC* threshold, *B*'s accumulated capital will be returned to them without penalty. There will be some amount of time ($1/\delta$ periods) in which the invested capital remaining in the business is not earning its cost of capital, but as long as the depreciation rate is high enough, the excess returns of the first epoch will outweigh the insufficient returns of the second and third epochs. (During the second and third epochs, *industry* growth will not be very high, because Firm *A* will be investing to replace the capacity that *B* is liquidating. Hence prices will not fall very fast.)

A will grow at its maximum rate every period until the fourth epoch of competition. It will be able to pay dividends only when (1) its competition has exited the market, and (2) prices have reached their long-term equilibrium. However, as long as *A*'s *ROIC* is higher than the cost of capital, *A*'s investors will be happy to reinvest their profits. And under the (admittedly overly simplistic) assumptions of base case, *A*'s *ROIC* will be above the cost of capital forever!

6. Agency, Foresight and External Finance

We now consider three variations on the base case corresponding to interesting problems that arise in practice. We first consider the problem of *agency*, when the managers do not follow a value-maximizing decision rule. Second, we look at what happens when investors have *foresight*, i.e., they can predict how the industry will evolve. Third, we assume that both firms have access to *external capital*, and ask if they will use it. In the analysis of foresight and external capital, we will look at outcomes with and without agency conflicts.

6.1 Agency

In the base case, we assumed that both firms obeyed myopic value-maximizing decision rules. They invested in future growth if and only if the prior period's *ROIC* was greater than the cost of capital. Implicitly this meant that the managers of each firm were *perfect agents* of the investors. Quite often, however, managers' interests are best served if *their* firm survives and grows, even if the capital markets offer more profitable opportunities to investors. This is a classic agency problem (Ross, 1973; Jensen and Meckling, 1976; Furubotn and Richter, 2005).

To model the agency problem, let us assume that the managers of both firms are indifferent to the investors' opportunity cost of capital. As a result, they will invest in new capacity as long as they have cash to do so. For ease of reference, we will call such managers *empire builders*. Empire building managers will not stop investing when $ROIC_B(t) < \rho$: they will continue to invest all the firm's earnings and grow at its *ROIC*. Hence the second epoch of competition will be marked, not by *B*'s withdrawal, but by *B*'s continuing growth and commensurately falling prices.

As long as $ROIC_B > 0$, Firm *B* will have positive profits, and money to finance its growth. But from its investors' point of view, it will be destroying value. By definition, *A* has a higher *ROIC*, thus its growth may be profitable even if *B*'s is not. But if *A*'s managers are empire builders like *B*'s, they will follow them down the demand curve, and both firms may end up destroying value.

Where will this end? Once prices fall to the point where *B* becomes *unprofitable* ($ROIC_B(t) < 0$), *B* must stop growing for lack of funds. Furthermore, it will need to use some of its

depreciation cash flow to pay for its deficits. (Depreciation is a non-cash expense thus a source of cash.) Thus eventually *B* will shrink, while *A* can continue to grow. At some point, we assume, *B* will no longer be viable and will exit the market. Thereafter *A* will be unopposed in the marketplace and can set prices either as a monopolist or to deter entry.

Thus in the presence of agency conflicts, the industry will follow the same dynamic pattern as in the base case except that the early growth phase will last longer. However, because *B*'s managers "don't know when to stop," there will be a period of value destruction for its investors. And depending on the magnitude of *A*'s *ROIC* advantage, the value destruction may spill over to *A* as well.

An interesting variant of this scenario arises when *A*'s managers are perfect agents and *B*'s are empire builders. As above, *B* will continue to invest after $ROIC_B < \rho$, thereby driving the price down. As the price falls, however, *A*'s *ROIC* may drop below the cost of capital, at which point (since its managers are perfect agents), it will stop growing and begin to disinvest. Then, one of two things can happen: (1) *B* may drive *A* out of the market (even though it has a inferior *ROIC*); or (2) *A*'s and *B's* investments and capacities may oscillate so that both firms survive. In the latter case, *A* will make approximately the cost of capital over the long term, while *B* makes a return that is above zero, but below the cost of capital. *B* will not be prosperous, but as long as its average *ROIC* is positive and there is no way to oust its empire-building managers, it will not go out of business.

6.2 Foresight

In the base case, we assumed that the firm's investors made their initial investments without understanding the industry's competitive dynamics. We now assume that investors both understand the game and know whether the managers are perfect agents or empire builders. With such knowledge, they can forecast industry dynamics. Will their superior understanding change their initial investment decisions?

Not surprisingly, the answer depends on what type of managers are in charge. If the managers of both firms are perfect agents (as in the base case), then, as we saw in the previous

section, Firm *B* will earn more than the cost of capital for some period of time, and when its *ROIC* falls below the cost of capital, *B*'s managers will stop investing, pay dividends, and gracefully exit. With full foresight of these events, *B*'s investors would still make the initial investment. *A*'s investors do even better, hence they would invest, too.

If both firms' managers are empire builders, *B*'s investors will lose their initial capital, for its managers will take every dollar of cash flow and use it to grow (in the early stages of industry evolution) or cover operating losses (in the later stages). And *A*'s managers will not withdraw, even if *B*'s actions drive their firm's *ROIC* below the cost of capital. Knowing what is bound to happen, *B*'s investors will not invest! But then *A*'s investors will be even better off than in the base case, because they will never face competition from *B*.

Things are more complicated if *A*'s managers are perfect agents, but *B*'s are empire builders. Then, if the two firm's turnover ratios, κ, are close enough so that for some range of prices:

$$ROIC_A = \frac{(1-\tau)\cdot[p^*(t)-c]\cdot K_A(t)}{I_A(t)} < \rho \;;$$

and:

$$ROIC_B = \frac{(1-\tau)\cdot[p^*(t)-c]\cdot K_B(t)}{I_B(t)} > 0 \;;$$

the inferior firm (*B*) can drive the superior firm (*A*) out of the market. However, because *A*'s managers are perfect agents, *A*'s financial returns will still be higher than the cost of capital in (almost) every period. Thus even knowing the outcome, *A*'s investors would still make the initial investment.

If *B* drives *A* out of the market, its returns will have the following pattern:

- first a period of profitable growth ($ROIC \geq \rho$);
- followed by a period of unprofitable and slower growth, with $ROIC < \rho$ (as *A* shrinks);
- followed by a period of no competition.

The attractiveness of *B* as an investment depends on the balance of returns in these three periods. It is possible that the positive excess returns in the first and third periods will outweigh the negative excess returns in the second. In this case, investors with foresight would want to fund *B*, even though

its managers are empire builders *and* it faces an invested capital disadvantage.

Table 1 summarizes these results. The first row shows what happens when investors are myopic. In this case, both firms will be funded regardless of any agency conflicts. *Ex post*, however, if both firms' managers are empire builders, *B*'s investors will lose everything. *A*'s investors will fare better, but depending on how long it takes *B* to die, they may not make much money. Finally, if *A*'s managers are perfect agents, but *B*'s are empire builders, then *A*'s investors will make money, but *A* may be forced out of the market by the more aggressive *B*.

Table 1
The Impact of Agency Conflicts and Investor Foresight on Industry Structure and Dynamics

		Managers		
		Perfect Agents	**Empire Builders**	**A-Perfect Agents B-Empire Builders**
Investors	**Myopic**	A, B funded; Both make money	A, B funded; B loses money	A, B funded; A makes money; B may overturn A
	Foresight	A, B funded;	A funded; B not funded	A funded; B may be funded (if it can overturn A)

The second row shows what happens if investors have foresight. *A* always gets funded. *B* gets funded if both sets of managers are perfect agents, and does not get funded if both are empire builders. Finally, if *A*'s managers are perfect agents, but *B*'s are empire builders, then *B* may get funded in the expectation that by "not knowing when to stop" it will drive *A* out of the market.

An interesting corollary of this analysis is that *B's* investors may be better off with empire building managers than perfect agents (Fershtman and Judd, 1987). This result arises when (1) *A*'s managers are perfect agents; (2) *A*'s invested capital advantage is not too great— so that an aggressive *B* can drive *A* from the product market; and (3) the market is not contestable— so that the value of *B's* long-run excess returns exceeds the value of the negative excess returns incurred while *B*'s *ROIC* is less than the cost of capital.

6.3 External Financing

Last we consider the possibility that the two firms can can obtain external capital to fund their growth. Assuming that investors are myopic, either firm can issue new equity as long as its *ROIC* is above the cost of capital. A firm that issues equity in turn can exceed the growth limits of its internal funding sources. However, by Proposition 1, faster industry growth simply accelerates the decline in price and *ROIC*s. *Thus external finance destroys value for both firms*.

There is one and only one scenario in which raising external capital and growing faster than one's *ROIC* is part of a value-maximizing strategy. If the key decision-makers of *one* of the two firms believe that (1) their firm will be the long-run survivor; (2) the rival firm's managers will withdraw rather than fight for the market; and (3) the value of being the sole incumbent is very high, then it is rational to suffer losses in the short run in order to accelerate the arrival of the long-run. In this case, the industry payoff structure becomes a tournament (Frank and Cook, 1995; Aoki, 2001). There is one "prize"—long-term market dominance—and all resources are directed at winning the prize. If the key decision-makers at one or the other firm perceive the world this way, then they will sacrifice their firm's short-run profitability in order to increase their probability of winning and accelerate the date of their rival's withdrawal.

The crucial difference between a tournament and the other scenarios we have modeled is the value assigned to being the long-run industry survivor. If this value is perceived to be very high, racing behavior is rational, and external financing may help win the race.

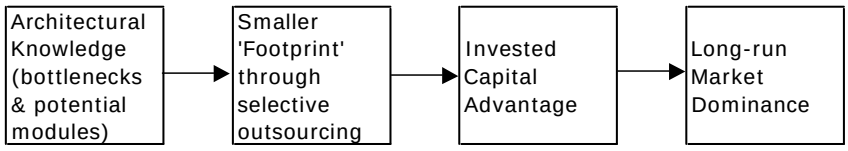
Alternatively, if the key decision-makers are empire-building managers, the terminal value of the firm does not matter. As long as the empire builders' firm survives, they have won. For them, external financing is a "no-brainer." This in turn means that foresighted investors cannot leave external financing decisions in the hands of managers who may be empire builders.

7. Empirical Evidence

In the previous sections, we argued that superior architectural knowledge of bottlenecks and

potential modules makes it possible for a firm to pursue a smaller footprint through selective outsourcing. The result of this strategy is an invested capital advantage for the firm with the smaller footprint. In the long run, subject to certain caveats, a firm with an invested capital advantage can drive its rivals out and dominate the product market. The links in our chain of argument are shown in Figure 1:

Figure 1
Causal Chain of our Argument



In the next two sections, we will support this argument with evidence from the field. Ideally we would like to verify each of the links in the chain. To do that, we must:

- pinpoint specific knowledge about bottlenecks and (new) modules in one firm but not its competitors;
- show that this knowledge enabled the firm to outsource activities that were insourced by competitors;
- verify that the firm in question had an invested capital advantage (a higher turnover ratio and ROIC); and
- show that the firm used its invested capital advantage to grow faster than its competitors and drive their *ROIC*s below their cost of capital.

We have two case studies which we believe support our theory. In the first—Sun Microsystems vs. Apollo Computer—we have evidence related to each of the four items in the causal chain. In the second case—Dell, Inc. vs. Compaq and other personal computer makers—we have evidence on the last three items, but no direct evidence as to the specific architectural knowledge Dell used to achieve its smaller footprint. Nevertheless, we feel that there is quite a lot of indirect evidence that Dell has pursued knowledge about bottlenecks and modules in its production and logistical systems, and applied this knowledge to the design of a smaller footprint.

Here is our evidence from the two cases.

8. Sun Microsystems vs. Apollo Computer[5]

8.1 The Product Market and Apollo

In the early 1980's, as the personal computer market was taking off, a smaller, but significant engineering workstation market arose as well. Developments in semiconductor memory and microprocessors, disk drives, and operating systems created the possibility of putting significant computer power—including graphics—on an engineer's desk. And with the emergence of low cost networking technology, it was possible to link the desktop machines into a network where engineers could share storage capacity and peripheral equipment and also communicate rapidly and effectively. All of this was possible at a fraction of the cost of a minicomputer—the original workhorse of engineering design.

Apollo Computer, founded in 1980, was the first to pursue this opportunity. It was soon followed by a number of competitors, including Sun Microsystems. Apollo and Sun both relied heavily on outsourcing. They based their products on Motorola microprocessors and bought (or had their customers buy) peripheral equipment including disk drives, printers, file servers, and monitors. They depended on third-party developers to write application software. And a large fraction of their sales were to original equipment manufacturers (OEMs)—companies that purchased systems, added applications software and specialized equipment, and then resold the systems often under their own brand names.

Apollo designed its workstations as an indivisible bundle consisting of hardware, a proprietary operating system (Aegis™); and a proprietary network management system (DOMAIN™). Each component required the other two in order to function. And to assemble its machines, Apollo built a continuous-flow manufacturing facility, specifically designed to make its workstations in high volumes. The factory was another part of Apollo's indivisible core architecture

[5] This case study is based on the following sources: Freeze and Clark (1986); Hall and Barry (1990); Soll and Baldwin (1990); Salus (1994); Zachary (1994), Gilder (1995); Garud and Kumaraswamy (1995); and Baldwin and Clark (1997a).

(Tushman and Murmann, 1998; Murmann and Frenken, 2006).

Many, perhaps most, practicing computer architects at this time believed that a high degree of integrality—that is, mutual customization and integration among the core elements of hardware and software—was necessary to achieve high levels of processing speed in a computer. It was all right to outsource CPUs, peripheral devices, and application software, but the core components of the machine needed to be interdependent in order to achieve the highest levels of performance. Apollo's proprietary operating system, Aegis, for example, was chosen over Unix (a choice actively debated within the company) because it offered significant advantages in networking functionality. (Aegis managed network resources so that any user could transparently access memory and processing power on other nodes and servers.)

Proprietary operating systems and networking software also created *user lock-in*—a loyal base of customers who would be the source of continuing revenue (Shapiro and Varian, 1999). For this reason, Apollo's use of proprietary components was applauded by investment analysts. The following comment is representative of Wall Street's view of Apollo in 1984:

> Its proprietary operating system will keep users with Apollo. The company has responded to Unix demand by running versions of Unix as a subset. Users will get more performance with Apollo's operating system and it keeps it from being a commodity product. Some users prefer Unix, and that segment Apollo won't get. But I don't see any problem; the market will grow 50% a year the next few years.[6]

In 1984 Apollo had 60% of a market that was growing at 50% per year. According to the conventional wisdom of the time, its position was enviable and its strategy impeccable.

8.2 New Architectural Knowledge

In the late 1970s and early 1980s, however, a few computer scientists began to look at the hardware-software interface of computers in a new way. Two leaders of this research effort were John Hennessy of Stanford University and David Patterson of the University of California, Berkeley. Hennessy and Patterson brought "a quantitative approach" to the field of computer architecture.

---

[6] Peter Labe, of Smith, Barney, Harris, Upham & Co., quoted in Baldwin and Clark (1997a)

They advocated measuring the performance of actual machines and breaking it down into mathematical components: instructions per program, clock cycles per instruction and seconds per clock cycle:

$$\text{Performance} = \text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock Cycle}};$$

Once one understood the performance of a given machine in this detailed way, one could spot the weaknesses in its architecture. Those weaknesses were precisely what we have been calling "bottlenecks"—places where the "common case" took too long to execute, and thus dragged down overall performance. Thus Hennessy and Patterson made Amdahl's Law a fundamental principle of computer architecture (Hennessy and Patterson, 1990; Patterson and Hennessy, 1994).

Throughout the 1980s, Hennessy, Patterson and others were actively working on architectural metrics and "reduced instruction set" (RISC) architectures. These ideas were in the air at the computer science departments of Stanford and Berkeley. Two of Sun's founders, Andreas Bechtolscheim and Bill Joy, came from this milieu.

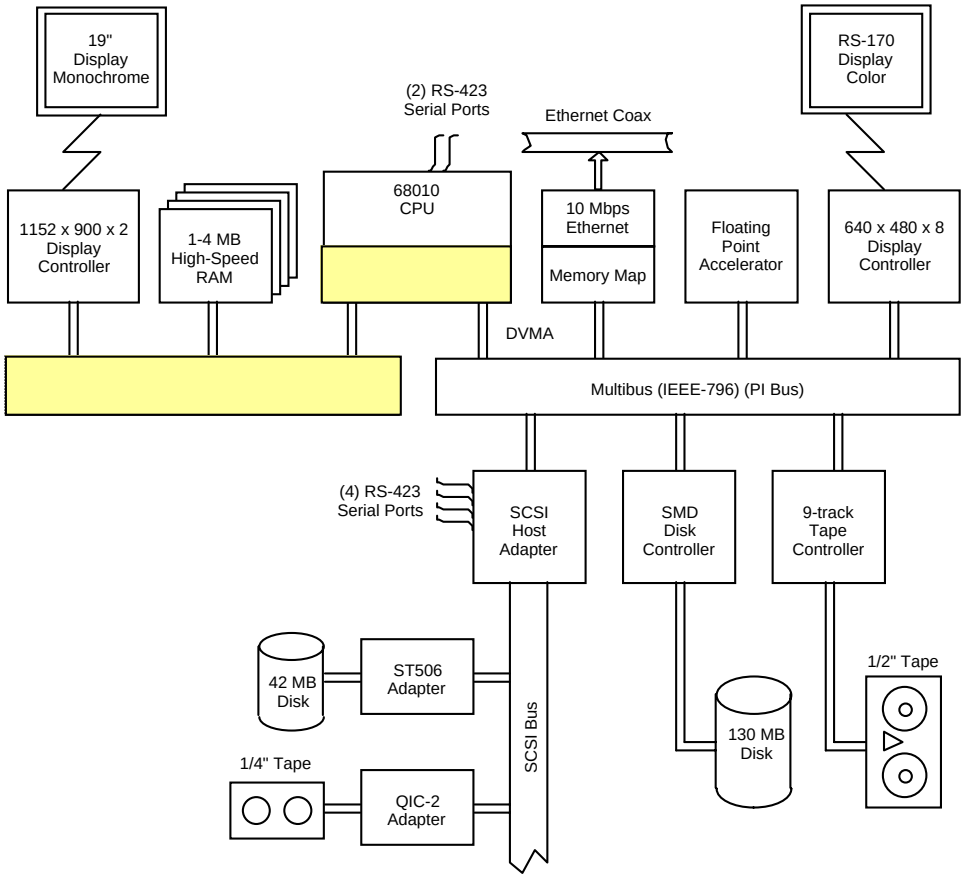
From the beginning, Sun pursued two historically conflicting objectives of computer design:

- low cost through the use standard interfaces and off-the-shelf parts; combined with
- superior speed and performance.

Sun's designers used their architectural knowledge of bottlenecks to get high performance out of standard parts. A case in point was the Sun 2, introduced in 1983, which was Sun's first major commercial success. Sun's architects had identified memory access as a bottleneck. As a remedy, they developed two proprietary hardware components. First, Sun designed and patented a special "no wait state" memory management unit (MMU) that eliminated many situations where the CPU had to wait to access memory. The second proprietary component was a high speed 32-bit internal memory bus, which connected the internal memory chips (1-4 MB of DRAM) and the video controller chips to the CPU. All other input-output operations used a standard Multibus setup. (Figure 2 shows a schematic diagram of the layout of the Sun 2. The special components used to solve the memory bottleneck are lightly shaded. Virtually all other elements in the diagram were standard components,

purchased from external suppliers.)

Figure 2
Schematic Diagram of the Sun 2 Computer Architecture



Source: Private communication, Andreas Bechtolscheim; This diagram appears in Baldwin and Clark (1997a).

The MMU and internal bus were crucial not only for fast and efficient memory-CPU coupling, but also for Sun's single board design. The MMU allowed Sun to avoid using cache memory (a special set of very fast memory chips designed to eliminate wait states): This not only reduced Sun's chip cost but also eliminated the need for an additional board. Thus, by effectively managing the CPU-memory connection, Sun achieved a "no-wait-state" design on a single board using standard parts.

Sun also invested in architectural knowledge about system software. William Joy, the main

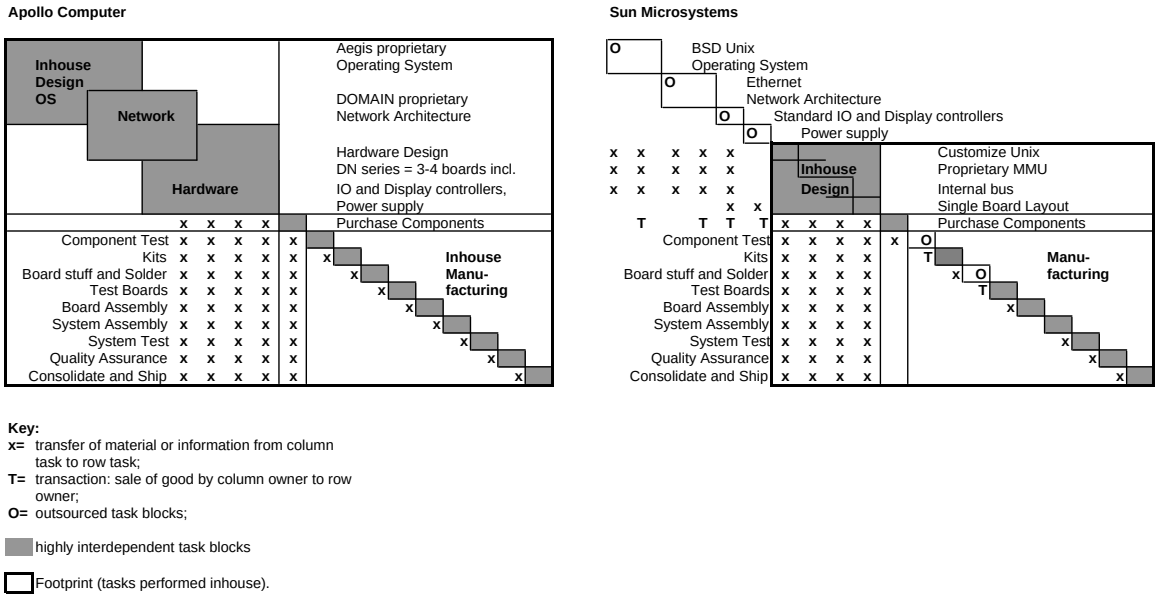
designer of the Berkeley version of the Unix operating system, was one of Sun's founders. He and the other Unix kernel designers employed by Sun understood Unix code at a deep level, and could advise the hardware designers on how to take advantage of its structure. These highly skilled programmers also exploited some of Unix's idiosyncrasies to enhance the machine's speed. For example, the operation of the no-wait-state MMU fit perfectly with Unix's conventions for establishing the hierarchy of operations (e.g., determining which address gets accessed first). The result was a meshing of the hardware and software that was most critical to the machine's performance. It was Amdahl's Law, "make the common case fast," brought to life in a real machine.

Finally Sun used architectural knowledge to redesign its production process. The single board design was crucial to its manufacturing strategy. Having only one board greatly simplified materials flow, reduced the amount of work-in-process inventory, made testing boards and systems much easier, and reduced the need for capital equipment and facilities. The overall impact was to substantially increase Sun's asset turnover relative to Apollo's.

8.3 Sun and Apollo Footprints

Sun's and Apollo's footprints can be represented using a task structure matrix (Baldwin and Clark, 2000, 2006). The rows and columns of a task structure matrix represent groups of tasks. An "x" denotes a transfer of material, energy or information from the column task to the row task. A "T" indicates that the transfer is also a transaction (column sells to row). Shaded boxes denote highly interdependent tasks and transfers. In Figure 3, the heavy black outlines indicate the footprint of each firm—the set of activities that took place inside each one.

Figure 3
Sun and Apollo Footprints Compared



Key:
x= transfer of material or information from column task to row task;
T= transaction: sale of good by column owner to row owner;
O= outsourced task blocks;

highly interdependent task blocks

Footprint (tasks performed inhouse).

Sun's footprint is much smaller than Apollo's. This is shown first by the smaller area inside the heavy border of Sun's task structure matrix. Apollo insourced critical design tasks, like coding the operating system, designing the network, and designing specialized hardware components. In contrast, Sun used standard solutions and commercial, off-the-shelf components. Sun also outsourced many steps in the manufacturing process, for example, component testing and board stuffing and soldering. Sun set up a build-to-order manufacturing process, in which kits of boards and components could be sent out for processing, then brought back and tested. Apollo's continuous-flow process required all steps to be completed in their own factory.

It is noteworthy that Sun's inhouse design process involved a highly interdependent core set of design tasks. At the heart of Sun's design, the processor, the memory, and the operating system were tightly linked to one another via the patented MMU, the internal bus, and specially crafted Unix code. However, relative to Apollo (and virtually all other computer makers of the time), Sun's interdependent design effort was *more narrowly focused* and *better targeted* on design elements that would make a real difference to performance or cost. Such fine-tuning, in turn, relied on architectural knowledge. Sun drew upon academic knowledge of the quantitative dimensions of instruction

processing and Joy's and other's deep knowledge of Unix. But it also developed internal knowledge of the precise bottlenecks in its own product designs and production processes.

8.4 Invested Capital Advantage to Sun

Above we argued that a firm with a smaller footprint based on superior architectural knowledge will have an invested capital advantage over its competitors. In a market with competitive (Bertrand) pricing, an invested capital advantage becomes an *ROIC* advantage. A firm with an *ROIC* advantage can in turn grow faster than its competitors, and may force them to leave the market.

Appendix A presents data on Apollo's and Sun's asset turnover, profitability and *ROIC* during the sixteen quarters for which we have comparable data.[7] It shows that Sun had consistently superior asset turnover and profitability, leading to a higher *ROIC* in every quarter.[8] The averages are summarized in Table 2.

Table 2
Average Asset Efficiency, Profitability and ROIC for Apollo Computer and Sun Microsystems Q2 1985 – Q1 1989

[7] This was the period ranging from one year before Sun's initial public offering to the time Apollo was purchased by Hewlett Packard.

[8] For purposes of these calculations, ROIC is defined as Net Income/Invested Capital. This definition can be trivially expanded as follows:

$$\text{ROIC} = \frac{\text{Net Income}}{\text{Sales}} \times \frac{\text{Sales}}{\text{Invested Capital}} .$$

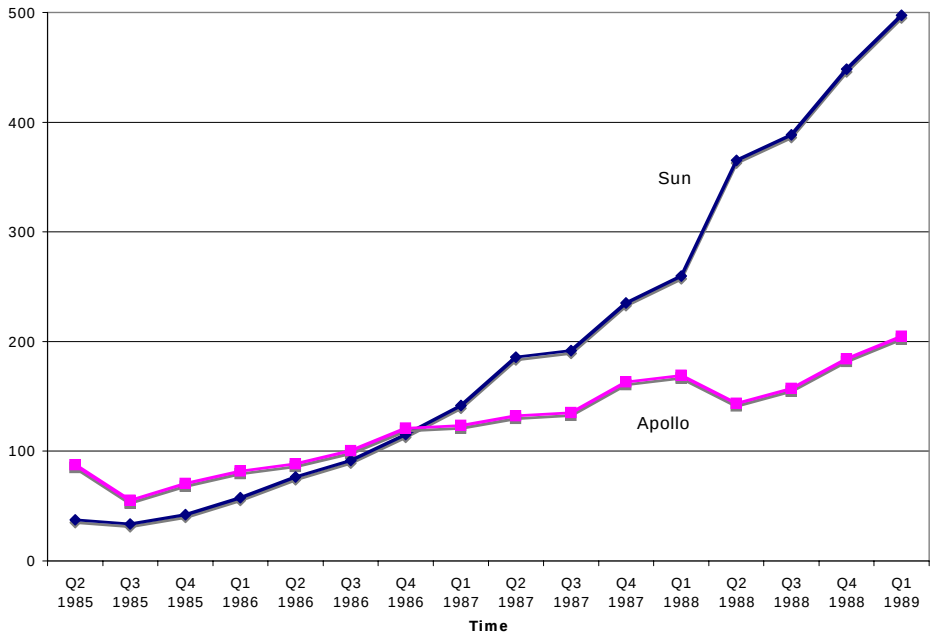
Thus higher profitability (the first ratio) and superior asset turnover (the second ratio) perforce lead to a higher ROIC.

Average over 16 Quarters:	Sun Microsystems	Apollo Computer
Asset Turnover		
Sales/Invested Capital (annualized)	3.33	1.86
Profitability		
Net Income/Sales	6%	2%
ROIC		
ROIC (excl Cash, Annualized)	20%	5%

Note: The 3rd quarter of 1985 was disastrous for Apollo for reasons that had little to do with Sun. We computed Apollo's ratios with and without that quarter, and found it did not significantly affect the results. The numbers for Apollo exclude this quarter.

Our theory of industry dynamics predicts that, in head-to-head competition, a firm with an invested capital advantage will grow faster than its competitors. Figure 4 shows the quarterly sales of Sun and Apollo from Q2 1985 through Q1 1989. Over these four years, Sun consistently outpaced Apollo in terms of growth. It went from being around half the size of Apollo (in terms of sales) to more than double.

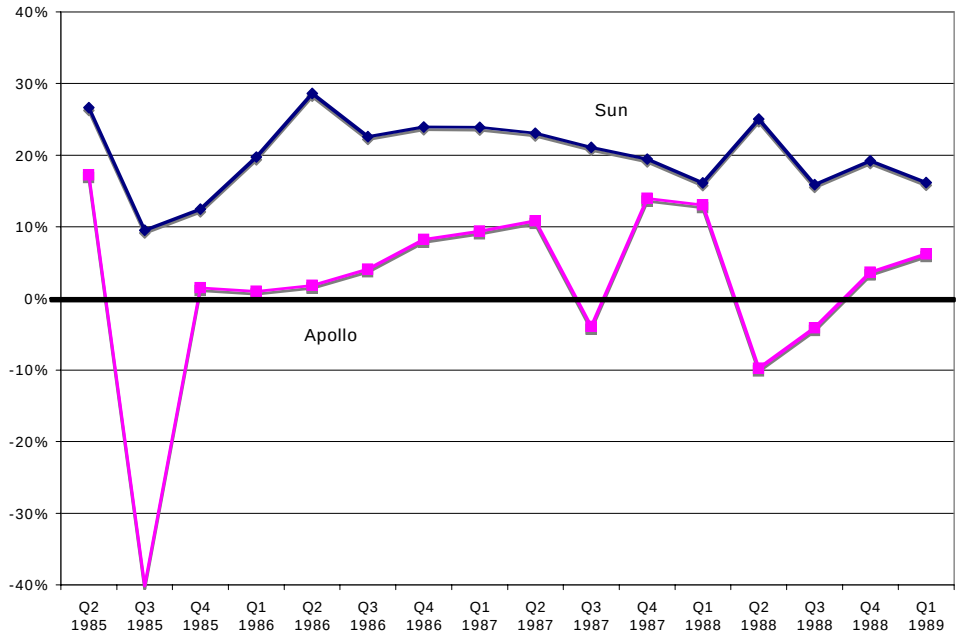
Figure 4
Sun's and Apollo's Sales Q2 1985 – Q1 1989



Our theory also predicts that a firm with an invested capital advantage will eventually set prices so that its rival's *ROIC* falls below the cost of capital. When Sun and Apollo were in head-to-head competition, 10-year Treasury bonds yielded from 7% to 10%. The companies' weighted

average cost of capital (comparable to their *ROIC*s) was in the range of 15%-25%. Figure 5 shows their quarter-by-quarter *ROIC*s (the figures have been annualized). After Q2 1985 Apollo's *ROIC* exceeded 10% in only 3 of the 15 quarters. Sun's own *ROIC* was in the low twenties and high teens. Thus, to a first approximation, Sun was earning its cost of capital, while Apollo was falling far short. At the time, Sun was viewed by knowledgeable observers as a ruthless competitor. It was rumored that no one except Sun was making money selling workstations.

Figure 5
Sun's and Apollo's ROIC Q2 1985 – Q1 1989



Finally our theory predicts that a firm with an invested capital advantage will use external finance only if (1) its managers are empire builders and/or (2) the value of being the long-run survivor greatly outweighs the short-term benefits of a high *ROIC*. Contradicting this prediction, Sun did not hesitate to go to the capital markets for financing. In sixteen quarters, Apollo raised $190 million in external finance, mostly in the form of debt while Sun raised $548 million mostly in the form of equity. And in Q4 1987, Sun entered into an unprecedented equity alliance with AT&T, which gave it rights to raise even more external capital whenever its managers felt the need.

The rest, as they say, is history. Although Apollo experienced good sales growth in the last half of 1988, its profitability and *ROIC* continued to languish. By the end of the year, its cash was down to $20 million; its debt capacity was exhausted; and the prospect of issuing equity was remote. In April 1989, Apollo was acquired by Hewlett-Packard (HP). The acquisition was touted as creating the largest engineering workstation company in the world, with a 30.4% market share vs. Sun's 28.3%. No one thought to ask whether the new business combination would be able to match Sun's

asset turnover and *ROIC*.

By all accounts, Apollo's acquisition did not fulfill HP's expectations. Over the next several years, HP gradually abandoned all of Apollo's product lines. Today Apollo hardware and software survive only in hobbyist computers.

8.5 Market Dominance?

Although Sun managed to drive Apollo out of the market (and put pressure on other competitors), it did not succeed in gaining dominance of an uncontested market. For one thing, Sun's other competitors—including IBM, DEC and HP— had deep pockets and in some cases were willing to lose money to protect their position in the workstation market. For another, much of Sun's architectural knowledge was not proprietary. In the 1990s, quantitative approaches to computer architecture and RISC-based instruction sets became common. (Hennessy and Patterson published two influential textbooks in the early 1990s.)

Finally, Sun's execution of its strategy was not flawless. In the second quarter of 1989, as Apollo was being folded into HP, Sun own internal processes descended into chaos. In 1989, Sun's ROIC averaged a paltry 4%, and its burn rate was over $100 million *per quarter*.

Why did Sun's managers play the game so aggressively? This question cannot be answered definitively by outsiders. But it is worth recalling that the company's key decision-makers were a team of young individuals with genuinely new architectural knowledge. They considered themselves to be strategic innovators as well as technological innovators (Hall and Barry, 1990). And for a while their strategy paid off: Sun made money and grew rapidly while its rivals lost money and struggled to keep pace with Sun's growth rate and technology.

Sun's managers, especially the CEO McNealy, also believed that economies of scale would eventually lead to industry concentration (Hall and Barry, 1990, p. 26). Thus in the eyes of its key decision-makers, Sun *was* in a tournament, playing for a small number of long-run industry survivor positions.

Finally in the eyes of Sun's managers, the alliance with AT&T was primarily meant to be not

a source of external funding, but the first tactical move in an audacious strategy centered on Unix. At the time AT&T owned the source code and trademarks of the Unix operating system. With AT&T's property rights and Sun's design capabilities, Sun's managers believed that Unix could become the dominant operating system for desktop machines and networks. After all, in 1988, Unix was scalable, cross-platform, and ran on networks, while its competitor, Microsoft Windows™, had none of these properties. Microsoft was then just beginning to develop Windows NT™, which eventually became a scalable, cross-platform, networked version of Windows (Zachary, 1994). But Windows NT did not ship until 1993, and even then was inferior to Unix in many ways.

However, bowing to pressure from a coalition of Unix users, including Apollo, IBM, DEC and HP, AT&T quickly backed away from Sun's aggressive strategy. And according to George Gilder, after the collapse of the Sun-AT&T alliance, Bill Joy sold a large block of Sun stock and used the proceeds to buy Microsoft shares, "thus becoming the second richest of Sun's four founders" (Gilder, 1995).

9. Dell Computer vs. Compaq (and Other PC Makers)[9]

Next we turn to the case of Dell. In the case of Sun, we had evidence on both the kind of architectural knowledge they applied and their architectural innovations. In the case of Dell, we know the company achieved a smaller footprint than Compaq and other PC makers, but we don't know exactly how they did so. Furthermore, a small footprint can be achieved in two ways: (1) through architectural innovation; and (2) through exercise of power in the supply chain.

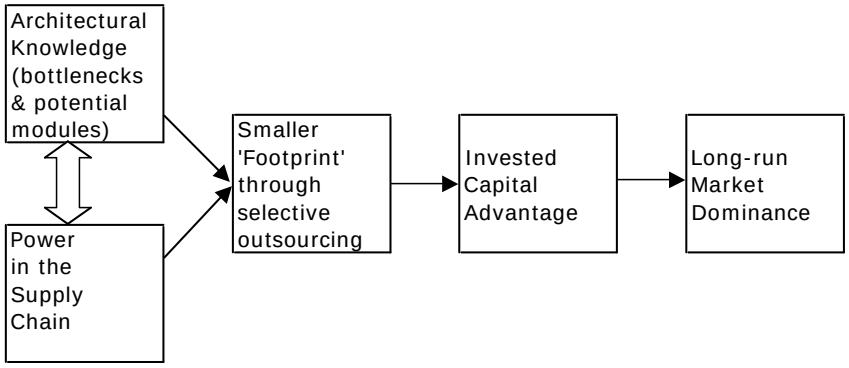
In practice, the two routes to a smaller footprint are complementary. Because the small footprint strategy requires outsourcing, it is helpful if the firm pursuing the strategy can negotiate from a position of strength. Indeed, more architectural knowledge makes power in the supply chain more valuable and vice versa—the two properties are formal complements in the sense of Milgrom

[9] This case study is based on the following sources: Baldwin and Feinberg (1999); Park and Burrows (2001); Shook (2001); Breen (2004); Holzner (2005); and Vance (2006a,b).

and Roberts (1990). In practice, complementary properties are likely to appear together, and their effects will be difficult to disentangle.

A restated version of our theoretical argument is shown in Figure 6. Here architectural knowledge and power in the supply chain both contribute to the smaller footprint. The two-way arrow between architectural knowledge and supply chain power indicates their high degree of complementarity.

Figure 6
Revised Causal Model

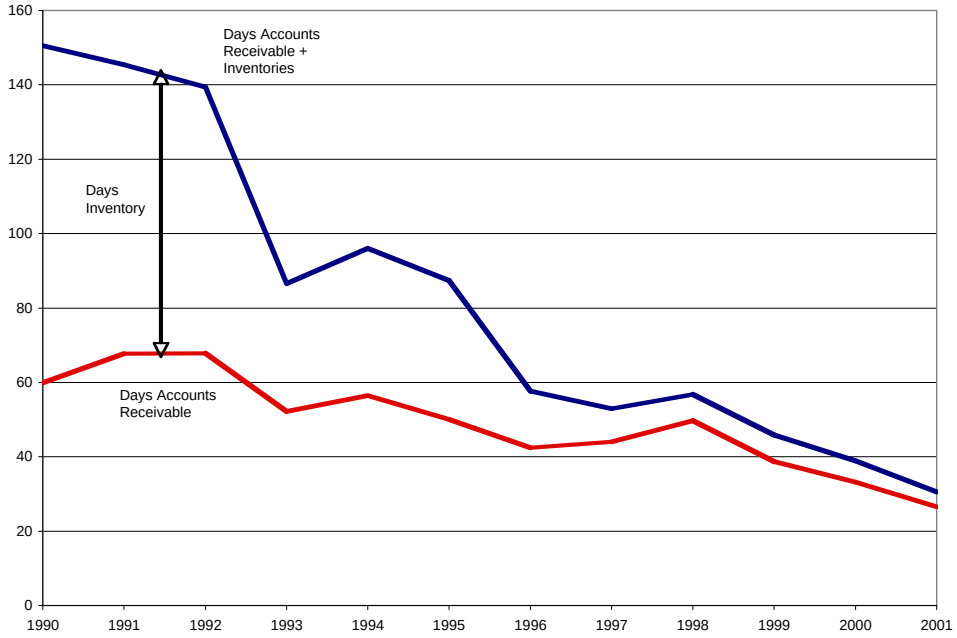


9.1 Dell's Footprint

Dell did not manage to achieve a small footprint overnight. In the early 1990s, Dell was one of a group of second-tier manufacturers of IBM-compatible personal computers (PCs). The largest stand-alone firm in this market was Compaq. Dell began to pursue a small footprint strategy in 1993 in response to a cash crunch. By the end of that year, its inventories dropped from 72 days to 34 days (see Figure 7).[10] But in 1995, Dell's sales grew by 52%, its invested capital increased by over $500 million, and it needed cash again. Its managers redoubled their efforts and brought inventories down to 15 days by the end of 1996. Combined with an increase in accounts payable (from 41 to 63 days) and a modest decrease in accounts receivable (from 50 to 42 days), this was enough to give Dell a negative cash cycle.

[10] Dell's fiscal year ends in the last week of January, while Compaq's ends on December 31. Thus it is appropriate to compare Compaq's 1995 financial performance, with Dell's *fiscal year 1996* performance. To avoid confusion, in what follows, we have relabeled Dell's fiscal years: thus "1993" equals "Dell FY 1994", "1994" equals "Dell FY 1995", and so on.

Figure 7
Dell's Days Accounts Receivable and Days Inventory 1990-2001



How was this reduction in inventory (and other assets) achieved? What distinguishes Dell is the incredibly tight integration—in both time and space— of the things it does inhouse: order-taking, assembly, and shipment (Fine, 1998). Reportedly, all these activities take place within two to five days, and sometimes in a matter of hours (Holzner, 2005). Dell doesn't own the components that go into its machines until the assembly process begins, and they bill the customer when the machine leaves the loading dock. If the sale is to an individual, his or her credit card is debited immediately, and Dell gets the cash. If the sale is to a large corporation, Dell may have to wait to collect. Averaging the two types of sales, Dell has about 30 days receivable at any point in time. But Dell demands more generous terms from its own suppliers: not only must they hold components until Dell needs them, but they must wait to be paid—on average 70 to 80 days.

Putting the pieces together—very quick inhouse manufacturing process, a relatively short collection period and a long payables period—the result is a *negative cash cycle*. This means that Dell receives cash for its machines before it has to pay for the components that account for most of the

machine's cost. In effect, Dell *borrows* components from its suppliers, builds machines, sells them, collects from its customers, and then *waits* to pay its suppliers. While it waits, it has the use of the cash it has collected. (Banks call this "float.")

How much of Dell's negative cash cycle is attributable to architectural knowledge? Dell's known practices include: (1) direct sales (no dealers); (2) requiring suppliers to locate warehouses close to their manufacturing plants and to hold inventory until it is needed; (3) locating manufacturing facilities close to the point of sale (i.e., North American orders are fulfilled by North American plants); (4) inhouse build-to-order manufacturing, based on proprietary software; (5) "cross-docking" of outsourced peripheral equipment;[11] (6) massive investment in an information infrastructure, including transparent, real-time sharing of key numbers with suppliers; (7) investment in intellectual capital as evidenced by patents; and (8) universal use of *ROIC* as a performance measure in all parts and at all levels of the organization.

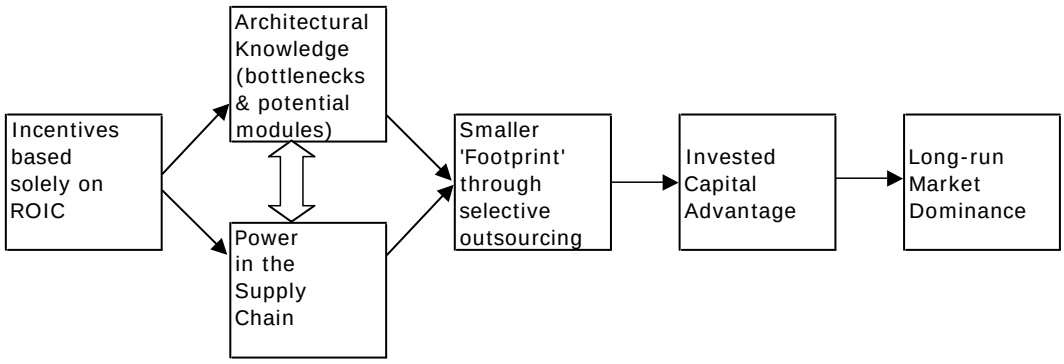
Of these practices, the first is a legacy of Dell's early history, and the second is evidence of Dell's power over suppliers. Practices (3) through (7), however—the location of manufacturing facilities, insourcing of manufacturing and the use of proprietary software to manage the manufacturing process, cross-docking, Dell's information infrastructure, and Dell's patents—are all evidence of superior architectural knowledge. Dell's outsourcing model is not the industry standard. Rather it is based on Dell's own remedies for the bottlenecks it has identified in order fulfillment, manufacturing and logistics.

Finally, practice (8)—the universal use of *ROIC* to measure performance—amounts to a meta-strategy for garnering and applying architectural knowledge within the organization. The value of knowledge and the merit of innovations at Dell are measured by their contribution to profitability and/or asset turnover. In effect, Dell's business model extends our causal chain one step backward as

[11] Cross-docking refers to the shipment of goods into and out of a Dell facility within a short period of time, without ever leaving the loading dock. Dell does not take ownership of the goods until they arrive, and bills the customer as soon as they leave. The practice results in very low levels of inventory for once the goods ship, they become accounts receivable. However, the practice requires very high levels of logistical coordination.

shown in Figure 8.

Figure 8
Dell's Business Model



9.2 Invested Capital Advantage to Dell

From the early 1990s through 2001, Compaq Computer Company was Dell's main competitor. Thus we can gauge the success of Dell's business model by comparing Dell's financial performance to Compaq's. Appendix B presents data on the two companies' asset turnover, profitability, *ROIC*, and cash cyle from 1990 through 2001. The averages are shown in Table 2. The averages reveal that Dell enjoyed a significant invested capital advantage over Compaq, but they do not show the trend. During this time period, both companies improved dramatically in terms of asset turnover, but Dell improved more. (See Appendix B.)

Table 2
Average Asset Efficiency, Profitability, ROIC and Cash Cycle for Dell and Compaq Computer 1990 – 2001

Average over 12 years:	Dell Computer	Compaq Computer
Asset Turnover		
Sales/Invested Capital	25.92	4.40
Profitability		
Net Income/Sales	5%	4%
ROIC		
ROIC (excl Cash, Annualized)	97%	19%
Cash Cycle		
Days AR + Days Inventory – Days Payable	20	76

The averages mask important trends in Dell's cash cycle. Beginning in 1996, Dell's cash cycle was increasingly negative, reaching –42 days in 2001. See Appendix B.

Again our theory of industry dynamics predicts that, in head-to-head competition, a firm with an invested capital advantage will grow faster than its rivals. This prediction is borne out in Figure 9, which shows Dell and Compaq's sales growth rates from 1990 to 2001. Once its new business model was in place, from 1996 on, Dell's growth rate consistently outpaced Compaq's. However, in 1996, Dell had less than one-third of Compaq's sales, and Compaq was able to grow very quickly as well. Thus, as shown in Figure 10, by 2001, (the last year Compaq was independent), Dell had caught up with Compaq, but not surpassed it.

Figure 9
Dell and Compaq Sales Growth Rates 1990 – 2001

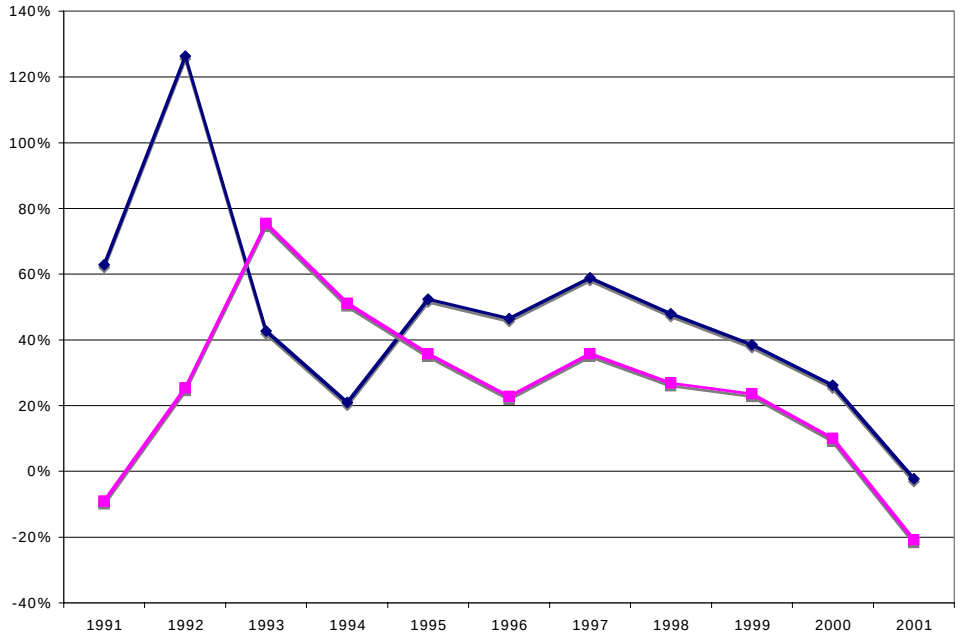
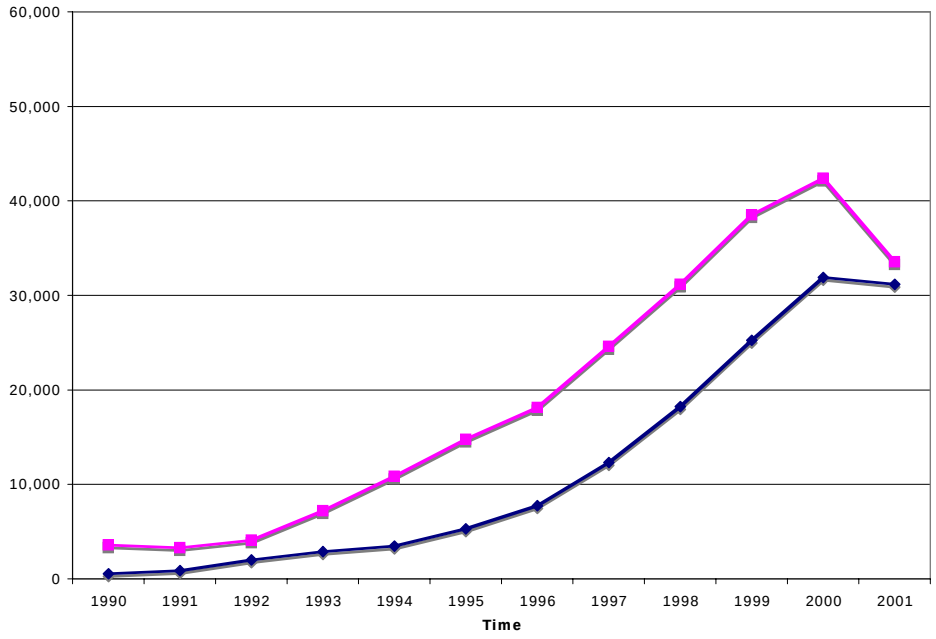


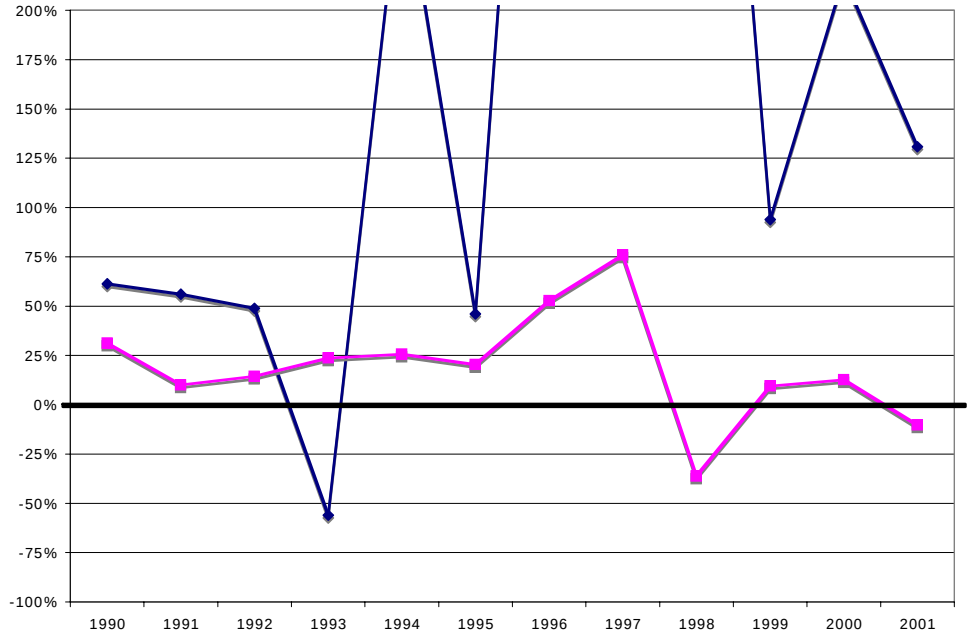
Figure 10
Dell and Compaq's Sales 1990 – 2001



It is also interesting to compare the two companies' *ROIC*s, as shown in Figure 11. From 1996 on, Dell's *ROIC*s were stratospheric (or in some cases not meaningful because its invested capital was negative). Compaq achieved respectable *ROIC*s in the range of 25% through 1995. Then as it began to pay attention to its asset turnover and cash cycle, its *ROIC* rosed impressively—to 53% in 1996 and 75% in 1997. But no matter how hard its managers tried, they could not match Dell's asset turnover and *ROIC*.

Figure 11

Dell and Compaq's ROIC 1990 – 2001



Our model of industry dynamics predicts that when two (or more) companies are in direct competition and one has an invested capital advantage over the other(s), both (or all) may grow profitably for a while. However, as the companies build their capacity and prices fall, there will come at time when the ROIC of the disadvantaged firm falls below its cost of capital. For Compaq, the day of reckoning came in 1998: after that year, its profitability never exceeded 2%, and its *ROIC* was consistently below its cost of capital and often negative (see Appendix B).

Unlike Sun, Dell did not make heavy use of external financing. In the first place, negative working capital is a source of funds, which can be used to purchase fixed assets. Furthermore Dell's asset-efficient business model meant that it did not have much to invest in—for example, it did not have to build new fabrication plants as a chipmaker would. Indeed in 1996-1998 and 2004-2005, Dell's *total* invested capital was negative: the fixed assets acquired did not use up the funds generated by its negative working capital. Proposition 2 implies that a company with negative invested capital has no *financial* constraints on growth. Its growth is limited only by (1) market demand and (2) organizational hazards.

On the other hand, a company with a negative cash cycle *cannot afford not to grow*. This aspect of Dell's business model is subtle and warrants further explanation. As a thought experiment, consider a company with a negative cash cycle of 42 days (this was Dell's performance in 2001). Assume the company collects $100 million today from machines shipped today. The cost of the machines is $90 million. Further assume that the company's sales will be 10% higher 42 days from now. When the company has to pay for the components shipped in today's machines, it will collect $110 million from the machines sold that day. Thus it can plan to pay today's suppliers with tomorrow's revenues and have money left over. But now suppose the business shrinks by 20% in the next 42 days. The company will then collect only $80 million, but will owe suppliers $90 million. Current revenue will then *fall short* of payables, and the company will need to raise an extra $10 million in cash to pay its suppliers. Thus for a firm with a negative cash cycle, *shrinking requires money*. If current customers don't provide enough money for payables coming due, then the money to pay those debts must come from other parts of the business.[12]

This means that, if demand softens, a company with a negative cash cycle will face a cash crunch. In contrast, a company with a positive cash cycle can shrink easily: as its sales decline, the cash generated from past sales will exceed the cash needed to pay current suppliers. For this reason, firms with a positive cash cycle may be inclined to keep their prices high during a downturn, while firms with a negative cash cycle will be inclined to keep their revenue up, even if it means lower (or even negative) profits per unit sold.

Based on this analysis, we predict that Dell will be inclined to drop prices aggressively during industry downturns. Indeed this is what happened in the first half of 2001. Early in 2001, industry sales slowed. In response, Dell dropped prices, and a full scale price war ensued. When the dust settled, Dell was still in the black, but Compaq and others were losing money and bleeding cash. It was rumored that the entire industry lost $1.1 billion in the first three quarters of 2001 (Park and

[12] Social Security works the same way: if the working population shrinks, current tax receipts will not be enough to fund obligations to retirees.

Burrows, 2001).

Compaq was a casualty of the price war: in early September, 2001, it was acquired by HP. In an echo of the Apollo acquisition, the HP-Compaq merger was touted as creating the largest PC company in the world. This time, however, many people questioned whether the business combination would create enough value to be competitive with Dell. The deal became the focus of an acrimonious proxy battle at HP. Three years later, Carly Fiorina, CEO of HP and architect of the merger, lost her job, in part because the merger with Compaq failed to meet expectations.

9.3 Market Dominance?

As Appendix B shows, Dell emerged from the industry downturn and its price war stronger than ever. Not only did Compaq and HP merge, so did lesser rivals Gateway and eMachines. And in 2004, IBM sold its struggling personal computer business to the Chinese PC manufacturer Lenovo. During this period, from 2002-2005, Dell delivered respectable growth rates (CAGR = 16.45%) and astronomical *ROIC*s.

Even so, the end result for Dell was not quite market dominance, as our simple model of industry dynamics predicts. Today in its traditional stronghold, the PC market, Dell is being challenged by very-low-cost offshore manufacturers, especially from Taiwan and China. And although it has expanded overseas, it is still mainly a North American company. Finally it "suffers" from having more cash than it can profitably invest. (In 2002, Dell began to repurchase significant amounts of its own stock, but cash still keeps building up on its balance sheet.)

Seeking new opportunities, Dell has set its sights on other markets: it now makes servers, notebooks, storage devices, and printers. It has had some success, especially in the server market. But, in the first two quarters of 2006, Dell's profits declined for the first time since the price war of 2001. And in stark contrast to 2001, when Dell's rivals suffered more than it did, this time, its competitors are doing well. Apparently they have managed to find market niches where an invested capital advantage is not the driving force of dynamic competition.

10. Conclusion

In this paper we laid out a dynamic strategy that can be employed by firms capable of architectural innovation. The strategy involves using knowledge of the bottlenecks in an architecture together with the modular operator "splitting" to shrink the "footprint" of the firm's inhouse activities. Modules *not* in the footprint are outsourced—module boundaries are redrawn and interfaces designed for this purpose. The result is an invested capital advantage, which can be used to drive the returns of competitors below their cost of capital. We explained how this strategy works and modeled its impact on competition through successive stages of industry evolution. We then showed how this strategy was used by Sun Microsystems in competition with Apollo Computer in the 1980s and by Dell in competition with Compaq and other personal computer makers in the 1990s.

References

Aoki , Masahiko (2001). *Towards a Comparative Institutional Analysis*, Cambridge, MA: MIT Press.

Baldwin, Carliss Y. and Kim B. Clark, (1997a) "Sun Wars: Competition within a Modular Cluster," in *Competing in the Age of Digital Convergence*, D. B. Yoffie, ed. Boston: Harvard Business School Press.

Baldwin, Carliss Y. and Kim B. Clark (1997b) "Managing in the Age of Modularity," *Harvard Business Review* Sept/Oct: 81-93.

Baldwin, Carliss Y. and Kim B. Clark (2000). *Design Rules, Volume 1, The Power of Modularity*, Cambridge MA: MIT Press.

Baldwin, Carliss Y. and Kim B. Clark (2006) "Where Do Transactions Come From? A Network Design Perspective on the Theory of the Firm," Harvard Business School Working Paper No. 06-051, available at http://www.people.hbs.edu/cbaldwin/ (viewed 7/29/06).

Baldwin Carliss Y. and Barbara Feinberg, (1999) "Compaq: The DEC Acquisition," 9-800-199, Harvard Business School Publishing Company, Boston, MA.

Baumol, William J. (1982) "Contestable Markets: An Uprising in the Theory of Industry Structure," *American Economic Review*, 72(1):1-15.

Baumol, William J., John C. Panzar, and Robert D. Willig (1983) "Contestable Markets: An Uprising in the Theory of Industry Structure: Reply," *American Economic Review* 73(3):491-96.

Bell, C. Gordon and Allen Newell (1971). *Computer Structures: Readings and Examples,* New York, NY: McGraw-Hill.

Breen, Bill (2004) "Living in Dell Time," *Fast Company*, 88(November), available at http://www.fastcompany.com/magazine/88/dell.html (viewed 7/29/07).

Engineering Systems Department Architecture Committee (2004) "The Influence of Architecture in Engineering Systems," Engineering Systems Monograph, MIT, Cambridge, MA (March); available at http://esd.mit.edu/symposium/monograph/ (viewed 7/29/07).

Eppinger, Steven D. (1991) "Model-based Approaches to Managing Concurrent Engineering" *Journal of Engineering Design*, 2: 283-290.

Fershtman, Chaim and Kenneth L. Judd (1987) "Equilibrium Incentives in Oligopoly," *American Economic Review*, 77(5): 927-940.

Frank, Robert H. and Philip J. Cook (1995) *The Winner-Take-All Society*, New York, NY: Free Press.

Freeze, Karen and Kim B. Clark (1986) "Sun Microsystems, Inc. (B)" Harvard Business School Publishing Division, Boston, MA.

Fixson, Sebastian K. (2005) "Product Architecture and Assessment: A Tool to Link Product, Process, and Supply Chain Design Decisions," *Journal of Operations Management*, 23(2005):345-369.

Furubotn, Eirik G. and Rudolf Richter (2005) *Institutions and Economic Theory: The Contribution of the New Institutional Economics*, 2nd Edition, Ann Arbor, Michigan: University of Michigan Press.

Garud, Raghu and Arun Kumaraswamy (1995) "Technological and Organizational Designs to Achieve Economies of Substitution," *Strategic Management Journal*, 17:63-76, reprinted in *Managing in the Modular Age: Architectures, Networks, and Organizations,* (G. Raghu, A. Kumaraswamy, and R.N. Langlois, eds.) Blackwell, Oxford/Malden, MA.

Gawer, Annabelle and Michael A. Cusumano (2002) *Platform Leadership: How Intel, Microsoft and Cisco Drive Industry Innovation*, Harvard Business School Press, Boston, MA.

Gilder, George (1995) "The Coming Software Shift," *Forbes ASAP* (8/28/95) available at: http://www.seas.upenn.edu/~gaj1/shiftgg.html (viewed 7/30/06).

Hall, Mark and John Barry (1990) *Sunburst: The Ascent of Sun Microsystems* Chicago, IL: Contemporary Books.

Henderson, Rebecca M. and Kim B. Clark (1990), "Generational Innovation: The Reconfiguration of Existing Systems and the Failure of Established Firms," *Administrative Sciences Quarterly* 35: 9-30.

Hennessy, John L. and David A. Patterson (1990) *Computer Architecture: A Quantitative Approach*, Morgan Kaufman Publishers, San Mateo, CA.

Holzner, Steven (2005) *How Dell Does It*, New York, NY: McGraw Hill.

Furubotn, Eirik G. and Rudolf Richter (2005) *Institutions and Economic Theory: The Contribution of the New Institutional Economics*, 2nd Edition, Ann Arbor, Michigan: University of Michigan Press.

Iansiti, Marco (1997) *Technology Integration: Making Critical Choices in a Dynamic World,* Boston MA: Harvard Business School Press.

Iansiti, Marco and Kim B. Clark (1993) "Integration and Dynamic Capability: Evidence from Product Development in Automobiles and Mainframe Computers," Harvard Business School Working Paper 93-047.

Jacobides, Michael G. and Sidney G. Winter (2005) "Co-evolution of capability and transaction costs: explaining the institutional structure of production," *Strategic Management Journal*, 26(5):395 - 413.

Jensen, Michael C. and William H. Meckling (1976) "Theory of the Firm: Managerial Behavior, Agency Costs, and Ownership Structure," *Journal of Financial Economics,* 3(4):305-360, reprinted in *Foundations of Organizational Strategy,* Harvard University Press, Cambridge, MA.

Kreps David M. and Jose A. Scheinkman (1983) "Quantity Precommitment and Bertrand Competition Yield Cournot Outcomes," *Bell Journal of Economics*, 14(2): 326-337.

Langlois, Richard N. (2006) "The Secret Life of Mundane Transaction Costs," *Organization Studies*, forthcoming. Working paper available at http://www.econ.uconn.edu/working/2005-49.pdf (viewed 1/22/06).

Langlois, Richard N. and Paul L. Robertson (1992). "Networks and Innovation in a Modular System: Lessons from the Microcomputer and Stereo Component Industries," *Research Policy*, 21: 297-313, reprinted in *Managing in the Modular Age: Architectures, Networks, and Organizations*, (G. Raghu, A. Kumaraswamy, and R.N. Langlois, eds.) Blackwell, Oxford/Malden, MA.

Milgrom, Paul and John Roberts (1990) "The Economics of Manufacturing: Technology, Strategy and Organization," *American Economic Review* 80 (3): 511-28.

Murmann, Johann Peter and Koen Frenken (2006) "Toward a Systematic Framework for Research on Dominant Designs, Technological Innovations, and Industrial Change," *Research Policy* (forthcoming).

Pahl, G. and W. Beitz (1995) *Engineering Design: A Systematic Approach, 2nd Edition*, London: Springer-Verlag.

Park, Andrew and Peter Burrows (2001) "Dell, the Conqueror," *Business Week* (9/24/01) available at http://www.businessweek.com/magazine/content/01_39/b3750039.htm (viewed 7/27/06).

Parnas, D.L., P.C. Clements, and D.M. Weiss (1985) "The Modular Structure of Complex Systems," *IEEE Transactions on Software Engineering*, SE-11: 259-66.

Parnas, David L. (1972b) "On the Criteria to Be Used in Decomposing Systems into Modules," *Communications of the ACM* 15: 1053-58.

Patterson, David A. and John L. Hennessy (1994) *Computer Organization and Design: The Hardware/Software Interface*, San Mateo, CA: Morgan Kaufmann Publishers.

Ross, Stephen A. (1973) "The Economic Theory of Agency: The Principal's Problem," *American Economic Review*, 63(2):134-139.

Salus, Peter H. (1994) *A Quarter Century of Unix*, Reading, MA: Addison-Wesley.

Samuelson, Larry (2004) "Modeling Knowledge in Economic Analysis," *Journal of Economic Literature* 57(2):367-403.

Shaked, Avner and John Sutton (1983) "Natural Oligopolies," *Econometrica* 51: 1469-84.

Shapiro, Carl and Hal R. Varian (1999). *Information Rules: A Strategic Guide to the Network Economy*, Boston, MA: Harvard Business School Press.

Shook, David (2001) "The Winner of the PC Wars: Dell," *Business Week* (5/1/06) available at http://www.businessweek.com/bwdaily/dnflash/may2001/nf2001051_655.htm (viewed 7/29/06).

Soll, Jack and Carliss Baldwin (1990) "Sun Microsystems, Inc. – 1987 (A), (B), and (C)" Harvard Business School Publishing Division, Boston, MA.

Steward, Donald V. (1981) "The Design Structure System: A Method for Managing the Design of Complex Systems," *IEEE Transactions on Engineering Management* EM-28(3): 71-74 (August).

Sutton, John (1991) *Sunk Costs and Market Structure: Price Competition, Advertising and the Evolution of Concentration*, Cambridge, MA: MIT Press.

Tirole, Jean (1988). *The Theory of Industrial Organization*, MIT Press, Cambridge, MA.

Tushman, Michael L. and Murmann, J. Peter (1998). Dominant designs, technological cycles and organizational outcomes in Staw, B. and Cummings, L.L. (eds.) *Research in Organizational Behavior*, JAI Press, Vol. 20.

Ulrich, Karl (1995) "The Role of Product Architecture in the Manufacturing Firm," *Research Policy,* 24:419-440, reprinted in *Managing in the Modular Age: Architectures, Networks, and Organizations,* (G. Raghu, A. Kumaraswamy, and R.N. Langlois, eds.) Blackwell, Oxford/Malden, MA.

Vance, Ashlee (2006a) "Dell: 'We'll Cut Prices and … er, Cut Prices," *The Register* (5/19/06) available at http://www.theregister.com/2006/05/19/dell_first_model/ (viewed 7/29/06).

Vance, Ashlee (2006b) "Dell Warns of Q2 Bloodbath," *The Register* (7/21/06) available at http://www.theregister.com/2006/07/21/dell_q2_down/ (viewed 7/29/06).

Williamson, Oliver E. (1985) *The Economic Institutions of Capitalism*, Free Press, New York, NY.

Zachary, G. Pascal (1994) *Show-stopper! The breakneck race to create Windows NT and the next generation at Microsoft*, Boston, MA: Little, Brown.

Appendix A
Sun's and Apollo's Comparative Financial Performance Q2 1985 – Q1 1989

	Average	Q2 1985	Q3 1985	Q4 1985	Q1 1986	Q2 1986	Q3 1986	Q4 1986	Q1 1987	Q2 1987	Q3 1987	Q4 1987	Q1 1988	Q2 1988	Q3 1988	Q4 1988	Q1 1989
Sales/Invested Capital (annualized)																	
Sun	3.33	4.77	3.28	3.12	3.36	3.72	3.09	3.24	3.32	3.94	3.13	3.27	2.94	3.62	3.00	2.92	2.60
Apollo	1.86	2.05	1.20	1.44	1.50	1.53	1.65	1.88	1.80	1.88	1.86	2.16	2.12	1.77	1.82	2.10	2.32
Net Income/Sales																	
Sun	6.06%	5.59%	2.90%	3.99%	5.88%	7.70%	7.31%	7.38%	7.19%	5.86%	6.75%	5.94%	5.50%	6.92%	5.29%	6.58%	6.24%
Apollo	2.41%	8.41%	-33.42%	1.04%	0.66%	1.18%	2.47%	4.36%	5.21%	5.75%	-2.12%	6.45%	6.16%	-5.51%	-2.28%	1.72%	2.68%
ROIC (excl Cash, annualized)																	
Sun	20.22%	26.64%	9.52%	12.45%	19.74%	28.63%	22.57%	23.93%	23.90%	23.08%	21.08%	19.43%	16.14%	25.03%	15.88%	19.23%	16.19%
Apollo	4.87%	17.26%	-40.16%	1.49%	0.98%	1.80%	4.06%	8.21%	9.39%	10.81%	-3.93%	13.95%	13.06%	-9.77%	-4.14%	3.62%	6.21%

Note: Apollo's averages exclude Q3 1985. Inclusion of this quarter does not significantly affect the results.

Appendix B
Dell's and Compaq's Comparative Financial Performance 1990 – 2005

	Average 1990-2001	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005
Sales/Invested Capital																	
Dell	25.92	11.62	9.47	8.98	58.63	59.11	8.50	NM	NM	NM	13.96	31.08	31.97	80.28	17.20	NM	NM
Compaq	4.40	2.19	1.88	2.25	3.25	2.95	3.36	6.75	9.22	4.38	5.10	6.32	5.19				
Net Income/Sales																	
Dell	5.31%	4.99%	5.72%	5.05%	-1.25%	4.29%	5.14%	6.68%	7.66%	8.00%	6.59%	6.83%	4.00%	5.99%	6.38%	6.18%	6.39%
Compaq	4.01%	12.64%	4.00%	5.20%	6.42%	7.98%	5.35%	7.25%	7.55%	-8.80%	1.48%	1.34%	-2.34%				
ROIC (excl Cash)																	
Dell	96.89%	61.21%	56.06%	48.82%	-56.08%	274.49%	46.07%	NM	NM	NM	93.92%	216.76%	130.77%	485.03%	110.38%	NM	NM
Compaq	19.09%	31.09%	10.07%	14.28%	23.72%	25.56%	20.37%	52.67%	75.85%	-36.21%	9.50%	12.56%	-10.39%				
Cash Cycle (Days)																	
Dell	20	71	86	70	42	36	47	-5	-10	-6	-19	-23	-42	-40	-40	-45	-39
Compaq	76	111	116	130	103	128	103	42	16	47	33	48	35				

NM= Not meaningful. Applies to observations where Dell's Invested Capital is negative.