



COMPETITIVE ADVANTAGE

CHAPTER 2



Competitive Advantage

*There is a time in the affairs of men,
Which, taken at the flood, leads on to fortune.*

*In such a full sea we are now afloat,
And we must take the current when it serves,*

Or lose our ventures.

—Shakespeare

In order to speak coherently about competitive advantage, I must stand on the shoulders of a giant. Michael Porter's definitive work⁶ on the subject remains the Bible on this topic, so much so that *competitive advantage* and *sustainable competitive advantage* are now a permanent part of the world's business lexicon. In this chapter (and those remaining), I hope to provide some insights into broad trends in information technology (the world according to Leyva). I also hope to

show how they might impact a firm's ability to gain and sustain a competitive advantage. Unfortunately, I often find it difficult to speak without a white board, so I will enlist the assistance of a number of my favorite graphics. Bear with me, I promise to do my best to keep you from snoring.

Two Tears in a Bucket

Goddamnit! I don't curse. I just use some words as adjectives.

—Dwight D. Eisenhower

Before moving on to *serious* matters, I must address an issue that is near and dear to my heart. I sent the first chapter of this book to my inner circle of friends and colleagues. Many responded with enthusiasm (probably out of kindness). The one consistent criticism, or item of commentary, that they all voiced was the political incorrectness, or inappropriateness, of the profanity I use in this text. I asked them if I had made my point, and there was universal agreement that I had. I asked if I had made them laugh. Again they conceded that I had. I asked if it sounded like me. They responded with more laughs and confirmations. Then I told them to fuck off. (Just kidding.)

So, what does this have to do with competitive advantage? Plenty. I believe that, apart from brilliant and clever strategies, competitive advantage can only be sustained, going forward, by having an honest conversation with the marketplace. This point is made most eloquently by a group of individuals who wrote a wonderfully human book called the Cluetrain Manifesto. Check it out at www.cluetrain.com, or better yet, go to Amazon and buy a copy for yourself, your friends, your kids, and for any other human being that you care deeply about. (No shit.) Here's a quote from the book:

We are not seats or eyeballs or end users or consumers.

We are human beings and our reach exceeds your grasp. Deal with it.⁷

Okay, Carlos, let me get this straight. In order to attain competitive advantage, you have to be willing to say *Neo-Nazi motherfucker* in print? No, not exactly. But

you must be willing to have an honest conversation. I wanted to write this book for my friends and colleagues in the technology trenches. I wanted to tell these stories in my own voice. I wanted it to sound like the stories I tell when shooting the shit and drinking a few beers at the local dive. Yes, I also wanted to be taken seriously professionally, but not at the expense of sounding phony.

An individual who hired me in a consulting capacity, as Chief Architect (in charge) at Big Automation Company, recently reminded me of a comment that I had made to him regarding his persistent offers of a permanent gig. I told him that if I took a permanent gig, I would be less inclined to tell him, “Fuck you, asshole. You’re full of shit,” as I have, from time to time, behind closed doors.

The point is that I am not knowingly going to help you do something that I believe to be absurd, or just flat wrong—not for all the tea in China. I am not going to lie to you or otherwise misuse your trust for personal gain. That is part of my value proposition. That is part of what I bring to the table. If you want someone to kiss your ass, then don’t enlist my help, or that of my company. If you are looking for an extremely capable individual who has (not often enough) the occasional flash of brilliant insight, then I am probably one of the better candidates that you are *ever* going find. However, I must warn you, I will probably use the word “fuck” (*usually* not at inappropriate times, however), as well as all of its wonderful derivatives.

A Moral Dilemma

I spent three years running the local professional services organization of Northwest Computer Company. I learned more about *business* (as opposed to technology) during this time, than in all of my previous years combined, or since. I have wonderful memories and enduring friendships from this period in my life. However, life at Northwest Computer Company caused me unmitigated stress and took a serious toll on my health. There were several reasons for this, many of which were of my own making. I will share a particular dilemma that troubled me beyond belief, and which is relevant to this discussion of competitive advantage and its sustainability.



A Moral Dilemma

In the early 1990's, Northwest Computer Company was the shit. They had managed to string Intel processors together (using their flavor of Unix) in a scalable architecture that was more compelling than anything else on the market. By the time I arrived on the scene in 1995, Sun, IBM and HP were whacking them upside the head. They responded with a number of clever, but ultimately unsustainable, strategies.

The first strategy was the result of a brilliant insight. The client-server revolution had left a gaping hole, with respect to the dearth of knowledge available in the marketplace regarding the new architectural paradigm. While their competitors were busy pushing iron, Northwest Computer Company radically revised their sales model to lead with architecture. The fact that some of their sales teams, and their nascent professional services organization, could speak to these issues was a significant differentiator that led to big wins. When this strategy hit the wall (the competition caught on) they discovered Data Warehousing.

Data Warehousing and Big Unix Iron

Data Warehousing was a client-server technology that required big Unix iron. The idea was to lead with professional services (data warehousing design and implementation) and bundle the iron as part of the deal (i.e. change the rules of engagement). The implementation of this strategy required more depth in their Professional Services (PS) organization. I was hired as a Project Manager in the local Houston office, and I was subsequently promoted to Professional Services Manager. The position was responsible for working with account teams providing domain expertise for pre-sales support and managing resulting PS engagements once a sale was concluded. The strategy was far more successful on the west coast and in the U.K., where the sales teams and the PS organizations *got it* and had the depth of talent to implement it.

Frankly, the strategy was broken for many reasons. However, the problem was primarily that it required a level of sophistication that was not universally available. Instead of selling a product, the sales teams were mandated to begin selling *solutions*. Never mind that this required a completely distinct mindset, one that the majority of account teams were totally ill prepared to market.

Many prospects, at least in the conservative oil patch, were suspicious. They quickly surmised that we were leading with PS in order to sell iron. By 1995, the Unix market had already begun a quasi-de-facto standardization process (to the degree that this was possible in the Unix space). Prospects were extremely reluctant to commit to a niche Unix vendor whose market share was decreasing visibly.

This dichotomy in marketing strategy resulted in a moral dilemma for me personally. I believed wholeheartedly in solutions. I believed that well executed solutions were in the best interest of our potential customers. However, our solutions, provided by our capable PS organization, could readily be implemented on our competitors' iron (often for less cost and reduced risk). In other words, I only partly believed in our value proposition, and from the point of view of Northwest Computer Company, not the most important part.

Fear and Greed

Knowing this in my heart of hearts, after three years, I *still* decided to accept a position as an account executive. Why? Because of ambition, greed, and stupidity (mostly stupidity). Account executives ruled the roost (position power), and rightfully so. Nothing in business happens until a sale is made. If you closed a few big deals, you could be in high cotton.

It was an outside long shot, but I was a born gambler. I was a kid who had grown up in the streets, and I figured that no matter what, I would always land on my feet. Not this time.

I had gotten damn good at technical marketing, could wax poetic in front of an informed crowd and be convincing, but marketing *ain't* sales. When you are selling someone a million dollar solution, you have to look 'em in the eye and convince them that this deal is in *their* best interest. I couldn't do it because I didn't believe it. And even if I could have, there was no way on God's earth that I was going to endure these long sales cycles. (A sales cycle of a year or more was not uncommon.) I didn't even like hardware!

I started feeling an impending sense of doom. I was going to fail at this and there wasn't a fucking thing I could do about it (or so I thought). I went into a severe state of depression and had to take a medical leave of absence for three months. I never went back.

Okay, Carlos, so what is the moral of the story? We already knew that you were a loser. The moral of the story is that honesty counts, professionally and personally. A disingenuous (at best) value proposition is not sustainable. Period. A bait and switch strategy won't work. The marketplace is much too savvy. At best, it will lead to Pyrrhic victories and short-term gains.

IBM is an example of a company that has apparently cracked the code. They are wildly successful selling both big iron and professional services. I have no special

insights into IBM, but they have most certainly managed to achieve success by having two totally separate business units.

Big Computer Company eventually bought Northwest Computer Company. For the talented employees of Northwest Computer Company, things did not work out so well. But they most certainly did for the shareholders.

Michael (Porter) is Still 'da Man

Michael Porter defines competitive advantage as follows (paraphrasing):

*Competitive Advantage is an activity-based theory of the firm. To compete in any industry, companies must perform a wide array of discrete activities, such as processing orders, calling on customers, assembling products, and training employees. Activities, narrower than traditional functions, such as marketing or R&D, are what generate cost and create value for the buyer; they are the basic units of competitive advantage.*⁸

I will attempt to highlight how information technologies can impact these activities in a manner resulting in distinct advantages. Porter's concept of *activities* is virtually synonymous with Michael Hammer's concept of *process*, as brilliantly put forth in *Reengineering the Corporation*⁹. This book rocked the business world when it was first released in 1994. It kicked off one of the largest monetary windfalls that the consulting industry has known in recent history.

Many would argue that the revolution failed. Certainly there were more than a few high profile failed projects to point to. But the same could be said about the Quality Movement, SAP, CRM, the Web, etc. The reason for most of these failures has more to do with our inability, from an organizational perspective, to adequately handle complexity and chaos than with the soundness of the underlying ideas and business premises.

The Usual Suspects Revisited

Another reason why these quasi-religious movements fail is because the *usual suspects* (and their poorer first and second cousins) pitch the “idea du jour” as if it were the Holy Grail. The search for the Holy Grail, and the implementation of the wisdom derived from it, always costs millions of dollars, involves a small army of people, requires a *program*, and is *strategic* to the survival of your enterprise. Nothing short of a complete overhaul will do. Talk about making mechanics and lawyers appear charitable!

However distasteful the results, we should be careful not to throw out the baby with the bath water. Much of what the business press subsequently labels as fads are actually important pieces of the puzzle. Put together enough pieces and perhaps you will gain some insights into the big picture.

The danger lies in simplistic thinking. There are no silver bullets. Implement every one, or any one of these *programs* and the results are almost guaranteed to produce anything but competitive advantage. Why is that? Because it seems obvious to me, to the point of absurdity, that a sustainable competitive advantage cannot be derived from something that your competitors can readily implement.

That is not to say there aren't good reasons to implement these solutions. If you had thirty-year-old accounting systems, then a SAP implementation probably made sense. If your “customer facing” processes and systems are outdated, then by all means, develop an ROI for a CRM implementation. Just don't be conned into believing that you are going to gain a competitive advantage. One could make a strong argument that these solutions are part of the cost of doing business, and nothing more. In order to derive a competitive advantage, you will need to add your own secret sauce, perhaps using these solutions as a launch pad.

Simply Profound

The beautiful thing about Michael Porter's analysis is that it is both profound and easy to understand at the same time. There are only two sources of competitive advantage:

- Become the low-cost provider.
- Differentiate your value proposition to the point where customers are willing to pay a premium for the added value.

That's it. There is another interesting variation, however: do one or the other within a specific industry segment.

So, which activities within the value chain should you focus your scarce information technology resources on? Focus on a limited set of activities that you believe will produce the desired advantage. You don't have the money or the time to focus on anything else. This is basic b-school stuff, although you wouldn't have known it by the craziness that went on during the dot-com implosion. Information technologies have the bewitching quality, when well executed, to allow the pursuit of both (low cost provider and differentiation) simultaneously, yielding advantages that have been rare to non-existent heretofore.

I will make a concerted effort to link strategy with implementation (the "what" and the "how"). This disconnect is a root cause of catastrophic information technology failures. Almost always, the players on both sides of the equation are disparate functional organizations and, like the U.K. and the U.S., are separated by a common language.

The dichotomy results in the *Blame Game* process pattern (see next chapter). Often the resolution of this dichotomy is referred to as *aligning the business strategy with the technology strategy*. Although this phrase sounds profound, it is much too broad and vague to be useful. The alignment must happen at a more granular level (the activity or process) to be actionable.

Again to borrow from ‘da man (M. P.):

Activities provide the bridge between strategy and implementation. When strategy was defined in terms of broad positioning concepts, a clear separation between strategy and structure was meaningful and useful. “What” and “how” were comparatively distinct. Armed with the recognition that the firm is a collection of discrete activities, however, previously common distinctions between strategy, tactics, and organization blur. Strategy becomes the particular array of activities aligned to deliver a particular mix of value to a chosen array of customers. ¹⁰

Specific activities within the value chain must become the targets of our information technology efforts.

The Second Law of Thermodynamics and Related Nonsense

Fasten your seat belts; this flight is likely to encounter some cognitive turbulence. This is where Spanish Harlem intersects with the second law of thermodynamics and information theory. You are about to enter the Leyva triangle, and there are no guarantees that you will emerge the same (or at all) on the other side.

Any serious student of software development is familiar with the term *Design Patterns* (singleton, model-view-controller, iterator, etc.). If not, then check out the book by the same name¹¹. At the risk of appearing academic (God forbid. I have a reputation to maintain), I am introducing the term *Process Patterns*. Process Patterns relates to the software development process *itself*, as opposed to specific design issues (i.e. artifacts that can be implemented in code).

Chaos and Recurring Patterns

I have observed a number of recurring patterns (usually after ingesting large quantities of peyote) that have an interesting and significant impact on our ability to develop systems better, faster and cheaper. There is nothing especially novel

about the term “Process Patterns.” Other authors have used the term “Rules of Thumb”¹² as a way of describing specific insights. Patterns are an integral part of chaos theory, and they convey information, allowing us to detect order within apparent randomness. Process Patterns attempt to do the same for the chaotic nature of software development.

What do the codes used for sending messages back from spacecraft have in common with genes or a molecule of DNA? How is it that the second law of thermodynamics, a physicist’s discovery, is related to communication, so that we can speak of the “entropy” of a musical score, or a page of text, or a conversation [or the software development process]? Why are knotty problems in the mathematical theory of probability connected with the way we express ourselves in speech and writing? The answer to all these questions is “information,” and the very fact that a single concept can link so many diverse ideas is an indication of its great generality and power.¹³

The introduction of chaos theory, information theory, and entropy is necessary because these fields of study contain the necessary theoretical framework(s) and lexicon for talking about something that is messy; random; complex; unpredictable; gnarly; and, more often than not, downright confusing. In other words, it gives us a way to discuss the software development process and the complex information technologies that result from it. The goal is to develop an understanding of the demons at work during the creation of software.

The premise is that information technologies will continue to be the prime mover in the endless pursuit of competitive advantage. If this is so, then all stakeholders have a need to become smarter faster. Process Patterns are a way of metaphorically speaking about the demons. While some patterns appear to name roles, they are intended to describe *behaviors*, as opposed to individuals. These behaviors either increase entropy (a bad thing) or decrease entropy (a good thing).

I will use the following useful, if somewhat simplistic, definitions:

- The state of things tends inexorably toward disorder and randomness.
- The measure of the disorder and randomness is *entropy* (the 2nd law of thermodynamics).
- *Chaos* is a state of things where chance is supreme.
- *Software development* is a chaotic process that, left to its own devices, will tend toward a maximum state of disorder.

Damn, the boys in the hood are wondering if I have completely lost my mind. That notwithstanding, we shall press forward and see if we can make some sense out of this techno-babble.

Decisions in Progress

The following equation, from Ludwig Boltzmann, a top gun physicist of German descent who lived in the 1880's, is engraved on his memorial stone in Vienna. And although amazingly simple, it lays the theoretical foundation for the rest of the chapter:

$$S = k \log W$$

“**S**” stands for entropy.

“**k**” is a universal constant known as Boltzmann's constant.

“**W**” has to do with the number of ways in which parts of a system can be arranged.

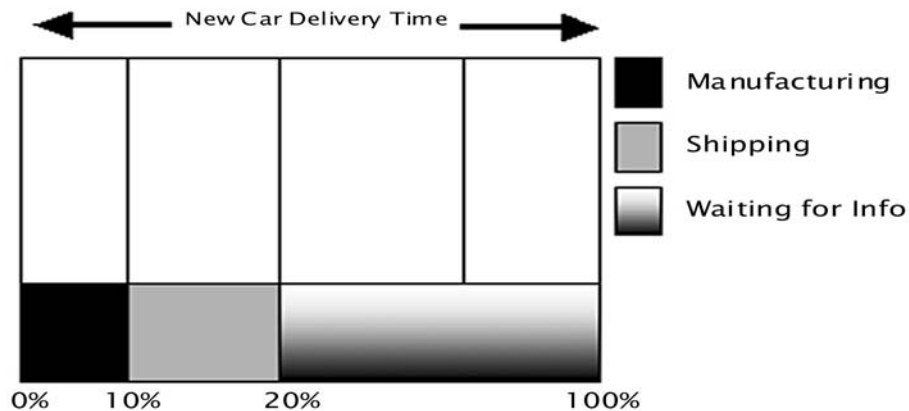
When starting development of any complex system, the number of possible ways that the system could be arranged probably approaches infinity (it is a really fucking big number). The “**W**” is quite large; therefore, the system is in a state of maximum entropy. Order is imposed, and entropy is reduced, by reducing the number of **decisions in progress** (DIP). By making intelligent, methodical and relentless decisions, constraints are imposed. As constraints are imposed, the

number of possible arrangements are reduced, order increases, and entropy decreases.

Competitive advantage, derived from applied information technologies, will be achieved by increasing the velocity by which the DIP yields to order. *Process Patterns*, or the lack of them, affect the DIP and have a powerful impact on whether your current project will actually contribute significantly to the end game.

Interesting Asides

Before proceeding with the *Process Patterns* discussion (see next chapter), I propose a slight detour in order to consider some interesting asides, and to clarify certain contextual matters. The following graphic depicts an interesting problem that the auto industry is faced with:



New Car Delivery Process

Eighty percent of the *new car delivery process* is spent waiting for information! The implication is that better information is required so that quality decisions can be made faster. In other words, in order to improve this process, the DIP must be reduced dramatically. Information technology has a role to play in this effort, but it would be naïve to treat this as an engineering problem. Why is that? Because

the critical part of the problem is almost certainly related to the number of players involved in the process and the fact that these players happen to be human beings. Human communication is chaotic and expensive. This was the *golden* insight that Fred Brooks had in *The Mythical Man Month*¹⁴ when he discovered that adding additional staff to an already late project would make it later.

Human Dialog as a Cost Factor

Human communication is expensive because human beings are messy; random; complex; unpredictable; non-linear; lazy (at times); and, more often than not, totally irrational. Language, as a communications protocol, is one of the *most* complex technologies on the planet. It is a beast that refuses to be easily tamed. How we communicate and collaborate with each other will need to improve dramatically if we are going to significantly compress the DIP within any non-trivial process.

Heuristics that improve interpersonal communications, within and across business processes, are as much, if not more, a part of the solution as *any* enabling information technology. The development of *Killer Apps* (application systems) will necessarily need to include knowledge from a number of different disciplines—including psychology, anthropology, linguistics and biology, to name a few.

Process Patterns aim to improve our ability to communicate *about* the software development process by illustrating behaviors that have a significant impact on our ability to execute. They are a set of mental models that attempt to provide a common language related to specific problems and issues. If you buy into the premise that human communication and dialog is an important aspect of the software development process, then purchase (and study) *Dynamics of Software Development*¹⁵ by Jim McCarthy. This book is filled with wonderful anecdotes and profound insights, told from an industry insider's perspective.

Commercial Software vs. In-House Development

I will use the example from the auto industry to help illustrate a subtle, but important, distinction between software development in commercial organizations, whose final product is *not* software (e.g. Ford Motor Company), and software development in the commercial software industry (e.g. Microsoft). Both use similar tools, methodologies, and human capital during the software development process; however, the organizational structures that drive the process are quite different.

IT departments tend to view their *market* as the internal customer organization (marketing, engineering, HR, etc.). It is a myopic view that tends to severely constrain potential solutions. A broader, more holistic, cross organizational worldview is required for efforts that are intended to create competitive advantage. Corporations have now entered the creativity business in a big way (yet most don't know it), and they must realize quickly that you can't get the genie back in the bottle (this is not your Daddy's Ford Motor Company). There must be a willingness to experiment with some of this *fuzzy logic* if progress is to be made.

Companies that produce commercial software on a large scale are usually organized in a manner that mandates, and facilitates, communication between senior management, marketing, program management, development, and R&D in order to achieve specific results in the marketplace. This communication, although not always optimal (depending on the set of players), is built into the software development process.

In companies whose primary mission is not software however, the communication process is often poorly defined, even dysfunctional, as it relates to software development. It also tends to be slower and less rigorous, negatively impacting both time-to-implementation and quality. There are a number of historical reasons for this, including the scarcity of institutional memory, as it relates to software development. The chapter entitled *Movie Making and Software Development*

includes a description of an organizational model that attempts to address these deficiencies.

Microsoft Goes to War

What would Microsoft do if they considered exploiting the *new car delivery process* for competitive advantage? Well, I can't say for sure, obviously, but I can surmise a damn good guess. First of all, their best and brightest would study the problem to determine if the resulting advantage would be compelling. The probability is quite high that they would crack the code and determine that the problem is about human communications as much as it is about technology. They would then weigh this project's potential advantage vis-à-vis their current portfolio of competition killer projects to determine relative importance to the organization. If, after this process, this project were at (or near) the top of the list, they would declare war; arm their best soldiers; pull together a top gun, cross-functional team; appoint a field general; and get after it with a vengeance!

A technology plan would be developed that included, at a minimum, what should be contained in the next couple of releases. All of this reduction in DIP activity would happen in an incredibly short period of time. No amount of resources would be spared until the goal was achieved. Top management would be all over the team's ass to produce. Why? Because they are at war! Most corporations are totally incapable of this type of mobilization.

The discussion of concepts such as the DIP, chaos, and the 2nd Law is not meant to undermine the importance of strategic thinking and raw technology. Both remain incredibly important. However, the message from this chapter is that while these are necessary, they are not sufficient. Corporations will only be able to sustain a competitive advantage by applying these amorphous ideas (e.g. reducing the DIP)—in conjunction with strategic thinking and enabling technologies—to a *set of activities* within the value chain in order to produce the desired results.

Hopefully, the upcoming *Process Patterns* discussion will help to clarify the fuzziness. Top decision makers can no longer abdicate responsibility for

technology decisions solely to the technology people. Technology is so tightly interwoven with the set of activities you are attempting to impact that it is difficult to separate the two. Learn to pay attention, or risk funding one disaster after another. An often-quoted definition of insanity is to “continue to do the same things over and over and expect different results.” Allow me to translate this into *Leyva speak* for you:

If you continue to use the same fucked-up processes, you will continue to achieve the same fucked-up results.

One additional point of clarification: corporations need not transform themselves into Microsoft in order to derive competitive advantage from applied information technologies. The ability to produce *insanely great* software is not required. The creation of damn good software, in conjunction with things that you currently do well, is all that is required. In fact, the resulting work product will be, more often than not, a *system* (as defined in Chapter 1) that produces great results, as opposed to a *product*.

That being the case, the organizational model that is required to produce software in-house does not have to possess all of the same characteristics that are found in successful commercial software organizations. However, there is much to be learned from *seriously* studying the best practices that have emerged from this industry. Keep in mind that you don’t have to be the fastest, or the first; you just have to learn to out run the competition.

*The ability to learn faster than your competition
may be the only sustainable competitive advantage.*

–Arie de Geus

Understanding Complexity

While it is certainly true that the tools and methodologies of the software development process have improved dramatically, it is equally true that the complex problems we are attempting to solve are growing faster than our ability to comprehend them. This is not a temporary aberration. It is axiomatic that our

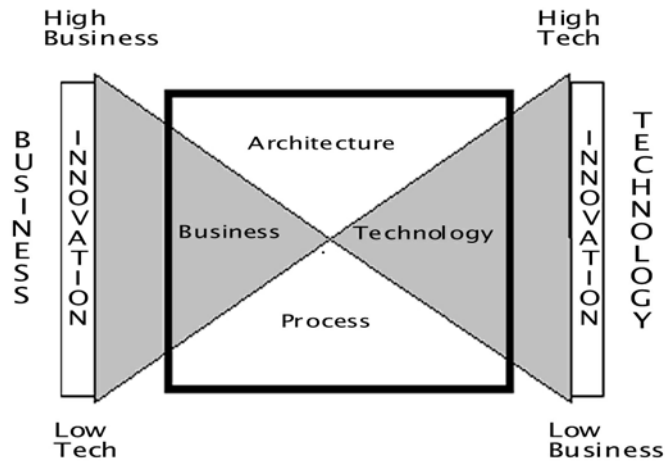
understanding will always lag behind our technologies. Therefore, progress cannot be made if we keep insisting on deterministic solutions. Progress will only be made when we begin to look for the deep order within chaos. That is, when we begin to recognize meaningful patterns.

The first chaos theorists, the scientists that set the discipline in motion, shared certain sensibilities. They had an eye for pattern, especially pattern at different scales at the same time. They had a taste for randomness and complexity, for jagged edges and sudden leaps¹⁶.

We must learn to amplify patterns that increase the probabilities of success, and minimize (or eliminate) patterns that increase entropy. Our sensibilities must become more attuned to the descriptive, rather than the prescriptive. This is out of necessity. It is the best that we can do (for now).

The “X” Diagram

IT organizations came into existence in order to provide a bridge between the business world and technologies that were beginning to have an impact on the business landscape. Initially, their role in life was to automate existing manual processes related to back-office systems (accounting, payroll, purchasing, etc.). I



The X Diagram

will use the “X” diagram to illustrate why IT organizations have had such a difficult time delivering viable technology enabled solutions. The rectangle depicts the solution space that was (is) IT’s purview.

As you recall from Chapter 1, in the days of IBM domination, all four aspects within the rectangle were essentially stable, certainly by today’s standards. However, with the advent of the destabilizing technologies previously mentioned, each aspect individually, and all of them taken together, began to go non-linear (i.e. they became chaotic). At the same time, IT was increasingly being asked to innovate, as well as automate.

The established order began to unravel, and it continues to reel from the onslaught of discontinuous change that wreaks havoc on its ability to cope. This technology stuff is no longer only an IT problem. It is an organizational problem. One that requires a radically different approach if we are going to leverage its benefits in a meaningful (and cost effective) manner.

We are never going to conquer complexity. All that we can do is become better acquainted with it and develop an eye for the jagged edges (and sudden leaps). We need to build into our processes an understanding that, at the inception phases of a complex system, we know very little about the end game. We have some vague ideas about the story we are creating, but only the creation process itself will reveal the true, final meaning.

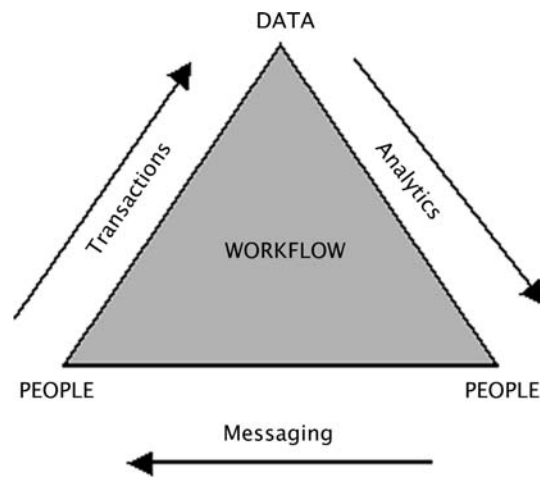
The whole question of complexity is one that has proved surprisingly subtle when investigated by modern theorists. Complexity is not just a matter of a system having lots of parts, which are related to one another in non-simple ways. Instead, it turns out to be a special property in its own right, and it makes complex systems different in kind than simple ones, enabling them to do things, and be things we might not have expected¹⁷.

It turns out that the whole is not only greater than the sum of its parts, but also quite different from it. Many of the eBusiness systems that we are currently developing are more complex than anything that we have attempted previously.

We must learn to eat the complexity elephant one bite at a time. The big bang theory is a recipe for disaster. Let me provide an example to help illustrate the complexity involved in such systems.

Workflows and eBusiness

The following graphic is a simple, yet representative, illustration of a typical workflow process within any organization. Transactions are captured by your front line data entry employees and deposited into your information systems as data. Knowledge workers subsequently massage the data and attempt to turn it into *information*. Information is subsequently routed to individuals who require it.



Workflow Processes

This type of workflow happens, in most organizations, via loosely coupled *manual* processes. Sure, individual components of the workflow (transactions, analytics, and messaging) are automated, but the workflow itself is not. Attempts to automate workflow have met with mixed (mostly poor) results. The low success ratio is due mostly to the complexity inherent in applying enabling technology

across internal organizational boundaries, for reasons that have been previously expounded upon (ad nauseum).

Now, consider what is involved in a typical eBusiness system. It usually involves applying enabling technology across corporate boundaries in order to reduce inefficiencies within the supply chain. In essence, workflows from different corporate entities are being linked together in what is now called a business web (b-web). There is no doubt that b-webs, enabled by web services technology, will have a powerful impact going forward.

But let us be clear about one thing: the complexity involved in making b-webs real is greater than anything we have attempted heretofore, notwithstanding compelling technology and vendor claims to the contrary. The technology offerings keep improving, and they should certainly be leveraged. But at the same time, we should recognize that most (not all) of the complexities lie within our processes and our abilities, as human beings, to effectively communicate.

Tear Down the Wall

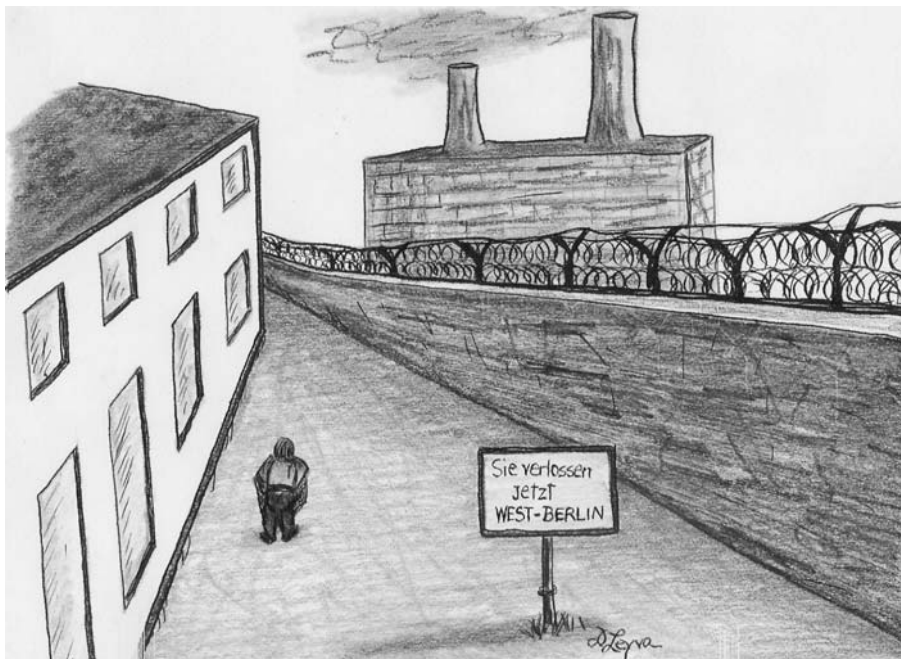
Strategy and implementation must exist as an integrated continuum if we are to gain a sustainable competitive advantage. If the strategy people throw their grand design over the wall to the technologists, it will get thrown back as a fucked-up solution, guaranteed! Michael Porter has *proven* that it makes no sense to speak of strategy outside the context of a set of activities within the value chain, just as it makes no sense to talk about reengineering the corporation.

How can you reengineer an entire corporation? What does it mean to do that? You can, however, reengineer a particular (limited) set of activities! Which ones? The ones that the people responsible for developing the strategy believe will result in an advantage. Which people? Ah, there's the rub. Why does the wall between strategy and implementation continue to exist? An obvious answer is that traditionally, the responsible parties have resided in separate organizational units. A more insightful answer is that some people (many people) prefer it that way. Why is that? Because, from the strategists' point of view, gnarly implementation

issues only get in the way of their wonderful thinking. Whereas, the technologists' point of view is "Don't bother me with strategy; it's not my job," which translates into "I need someone to blame" when (more often than "if") the project goes awry. In other words, the wall provides convenient cover for lack of accountability.

Progress is a nice word, but change is its motivator, and change has many enemies.

—John F. Kennedy



Tear Down the Wall

The Next Big Thing?

You must be kidding. There will always be a *next big thing*! It is a meaningless question. Hopefully, this is not the question that you had foremost in your mind at this point in the discussion. The question I would be asking is "What can I do

to improve upon this dreadful state of affairs?” Thanks for asking. The answer is simple, but its implementation is daunting: improve your organizational process literacy with respect to software development.

There is no question that software will become even more strategic going forward.

Much of a corporation's expertise and procedures are represented in its software. Corporate software is rapidly growing in complexity. Packages, templates, software objects, and automated development tools will allow increasing complexity as time goes by, encapsulating an ever-growing body of human know-how. Some corporate software has become a valuable strategic resource that could not be replicated quickly by competing organizations.

When American Airlines was highly profitable, its CEO, Robert Crandell, was asked whether he would sell the airline. He replied that given the choice of selling the airline or selling its software, SABRE (which links the airline to booking agencies worldwide), he would rather sell the airline; the software was more valuable.¹⁸

The remaining chapters in this book focus on heuristics that aim to improve corporate literacy in the software development process. The intended audience includes nearly all stakeholders within the firm, especially senior executives and information technology professionals.

Obviously, one need not become an IT professional in order to acquire the necessary literacy. However, a sustainable level of organizational literacy is required, since nearly everyone within the firm will be required to participate in these creative endeavors as we all build the software-centric enterprise. It is my firm belief that, for the foreseeable future, the software-centric enterprise will be the only type of enterprise that will consistently attain competitive advantage.