1

The DNA of Consistent Execution

Setting the Stage: Getting Stuff Done Quickly, Effectively, Consistently

Mike is senior vice president of retail banking applications for one of the world's largest financial services organizations. We met with him to discuss executional consistency—consistently performing at scale across an organization and in alignment with organizational goals. Actually, the meeting didn't start out to be about executional consistency. We were there to talk about the main topic of this chapter: challenges businesses face to *do stuff*—to execute, to perform, or to do whatever achievement verb best suits your environment. Jim Dillon, CIO of the State of New York, frames this topic well: "I'm happy to do more with more, or less with less, but I am not even going to talk about doing less with more." Aside from clever wordplay, there is deep understanding and insight expressed in Jim's phrase: We need to figure out how to get our work done more effectively. That's the focus of this chapter, and that was Mike's reason for talking with us. Mike wanted some perspective on the DNA of execution and the relationship of shared vocabulary and semantics to getting stuff done effectively. He wanted ideas on how to reduce time and expense he incurred getting things done; and he wanted to know about tools, methods, and techniques that might help him get "more done." Note that the conversation didn't start well. The actual conversation went something like this:

Mike: I don't have much time to spend with you, and if we're here to talk about methods and tools, I can tell you that everyone in my organization is trained and uses the Rational programming tools, and they use their Rational Unified Process for development. [Note: Rational and Rational Unified Process are a technical development environment and set of methods for doing technical modeling and analysis work.] In fact, I have more people using modeling tools right now than you have people in your entire company. So maybe we should have a cup of coffee, have a pleasant chat for 10 minutes, and then we can all get on with our day.

US: We'd prefer tea if that's possible [since tea takes longer to prepare]. But while we're getting ready to go, would you please tell us what the major challenge you face is around this large project you're undertaking? [Okay, so we're not fully on top of our grammar in early morning meetings.]

MIKE: Well, okay [he said, pouring his cup of coffee and pointing to the tea bags]. We have a hard time keeping up with the changes demanded by the business. I have over 150 different retail banking products and underlying applications here that support all aspects of the retail bank, and we can't change, test, and get things into production fast enough to keep up with new products and features that have to get out to the market.

US: [Now getting up and getting ready to go.] We see that problem over and over. Tell us, since you have a solid understanding of each of your applications and the products that you support, have you spent the time to look *across* your various applications and products to see if there are patterns where you can reuse things across applications?

MIKE: Hmm, great question . . . and please sit down. Give me an example.

US: Well, you're a retail bank with approximately 150 different products you offer to your customers. No matter how different they are, at some point, each probably requires a customer to open an account with the bank—or something like that. Do you have any sense of how many separate "Open New Account" functions you have scattered through the applications you maintain? Or for that matter, how many different ways the people in the different parts of the bank actually process opening a new account?

MIKE: [Reflecting . . .] No. . . . I *don't* know. But talk to me, what can I do that would help me?

US: Say that in your more than 150 applications there are 20 different places that "Open New Account" is required. We'd guess, based on what we've seen elsewhere and given that your company has acquired a number of other banks over the past few years . . .

MIKE: Yes! and I've gotten every one of their [profanity deleted] applications into my shop to support . . .

US: We'd guess that almost every one of those 20 "Open New Account" functions has been written uniquely, and probably each has buried in it business rules and business logic that you have to change each time the bank changes its rules, get new regulations that affect your products, or the bank changes or extends its product line.

MIKE: That's right.

US: Imagine the person-hours of maintenance you'd be able to use for other things if, instead of 20 "Open New Account" functions, you had a single function that all of your 150 products and underlying applications used.

MIKE: That would be a major savings . . . and there are lots of functions besides "Open New Account" that this would apply to. How would we get there?

US: Normally we'd tell you to start by modeling your business processes and applications—capturing the tacit intellectual property that's buried in the technology programs, the databases, and the heads of the experts who maintain them into higher-level abstractions of what they are and what they do—so that we could look for patterns of use and duplication. Given that you are using the Rational tools, you've already been codifying the information about your application portfolio into a form [models in Universal Modeling Language, or UML] that makes it more straightforward to find the patterns, figure out what is the same and what's different, and model a single function that could replace the 20 unique functions. We would start here, but equally if not more importantly, begin to understand what are the business processes that touch or are touched by these patterns. Knowing what touches what around these patterns will provide significant insight into what you can start to do, right away, and how to do it.

MIKE: All right, then [having asked us to stay for quite a while longer]. Let's start tomorrow on "Open New Account." By the time we've done that work, we'll have prioritized the next 10 business functions that we spend the most effort on, and we'll do the same thing for them.

Since that conversation, we found over two dozen functions—from the original "Open New Account" through "Research Account Dispute" and "Close Account." Each of these appeared in multiple business processes and the applications supporting the more than 150 and constantly increasing new products that were created or enhanced. Each of these as well were used by different groups using the applications that contained them worldwide throughout the retail bank. This one global retail bank—with a single global brand and an active campaign to ensure a great customer experience—actually looked like multiple, disconnected banks, none of which did business the same way. This lack of integration creates implications for many parts of the bank's business:

- *For the customer:* Often, when the customer wants to do more business with this bank, they are treated as though the bank knows nothing about them. This lack of a personal touch (or the illusion of a personal touch) means that the customer doesn't perceive any particular advantage in staying with this bank, and so is more likely to leave or is less likely to broaden his or her relationship with the bank.
- *For the employees who interact with the customer:* Since each part of the bank does things differently, an employee moving from one part of the bank to another—say from Mortgage Lending to Home Equity Lending—needs to learn new ways of doing things even though the businesses may be very closely related. This adds cost and effort to bank operations. Worse, it constrains an employee's mobility reducing their chances for new, challenging assignments and possibly making them more likely to look elsewhere for career opportunities.
- *For the employees who support the applications that support the employees who interact with the customer:* Many different implementations of similar but slightly different functionality create an environment that is prone to error, high pressure, and "death

march" types of projects. This leads to burnout and dissatisfaction in the very analysts and programmers who may be the only ones who know the quirks of some particular program implementing familiar functionality.

- *For the senior executives and CEO:* Frustration because the vision of one global brand and one customer experience—a vision that he or she articulated to the board, the analysts, and the shareholders three years ago—is still years away from reality due to the expense and the time eaten up by the lack of executional consistency in the business processes and the applications that support them.
- *For the shareholders:* The extra cost and the lack of agility due to the lack of executional consistency negatively impact the company's earnings and stock price.

In this chapter, we define the concept of *executional consistency* and analyze the DNA of executional consistency—the underlying characteristics and behaviors that enable an organization to get done what needs to get done quickly, consistently, and effectively. Once we have identified the needed characteristics—and underlying DNA for executional consistency—we explore in Chapter 2 some methods, tools, and processes that can be used to create those characteristics, and in Chapter 3 we examine some methods and processes to actually change—permanently—the organization to create the capabilities of getting done what has to get done, over and over again.

Extent of the Challenge: Why Is This *So* Hard?

Here's a question for you: Just how important is being able to execute—to align your objectives with your activities? This seems like a silly question. Of course it's critical. But as many of us know, it's extremely difficult to do and not many businesses know how to do it. Conflicting organizational activities, silos, redundant processes, and confusing, if not complicated, governance policies block effective, consistent execution. More than 64 percent of C-level executives from 250 midsized to large companies in the United States and the European Union have said that being able to execute, to "react quickly to changing business opportunities, models, technologies, and processes is critical for their success," and yet is nearly impossible to achieve.[1] Key to being sufficiently responsive is the capability to execute and align executive statements of what's supposed to happen with "stuff" that must be addressed to go in the direction at the speed and impact desired. The difficulty, as we all know, lies in actually *being able* to do so—to "do the stuff" that has to be done. What *is* it about an organization that makes execution so difficult? And what is common to these difficulties?

Common Challenges to Getting Stuff Done

Common challenges to getting stuff done can be sorted into the following categories:

1. *Getting your stuff out the door—a.k.a., Reducing your cycle time:* Whether regulatory-driven or a result of disruptive technologies or competitor moves, this particular challenge manifests itself in problems getting new products or services out the door fast enough. In public sector organizations, the challenge is how to become more responsive to ever-changing legislative demands for new services, or convert an agency or department into a more "constituent-driven" organization.
2. *Integrating existing with emerging technologies—a.k.a., Getting "value" from your IT activities:* Up to 70 percent of IT budgets are spent on maintenance and redevelopment rather than on new application development, and 60 percent to 80 percent of application functionality is redundant leading to a significant drag on cost levels. Companies spend a great deal of time and money on maintaining heritage or legacy applications for a good reason: They work! At the same time, the never-ending introduction of new technology and business functions means that new technical skills are required to complement the ones we already have. The tension between *what we have* and *what we need*

to incorporate is even further exacerbated by things like extensive consolidation, acquisition of companies into other companies, and/or the result of initiating multiple projects to meet multiple objectives multiple times over multiple years. The result: complex, different, and often redundant technical environments compounding the difficulty of getting different systems to talk with each other, as well as both figuring out how to evolve the skill-sets of your people, and figuring out effectively how to integrate existing with emerging technologies. And on and on it goes.

These first two problems have been an ongoing challenge for the past 30 years and, for that reason, we won't spend much time on them. The following two are becoming increasingly critical, warranting more discussion:

1. *Disconnected or disappearing intellectual assets—a.k.a.*, *Unlocking invisible value:* In too many organizations, the knowledge of how a particular process works, or of what an application actually does, is locked in the heads of one or a small number of individuals. In the next five years, more than 40 percent of the workforce in U.S. state and local government organizations will reach retirement age and retire with the knowledge that formed the "glue" that created whatever executional consistency exists in those organizations today.

The CIO of the Retail Banking Business Unit of a major global bank has identified *training* as one of his most strategic initiatives. He faces a crisis because much of what he called the "organizational wisdom"—the skills and know-how critical to keeping the business running, to develop, deploy, and maintain specific retail banking applications—was either "forced out or is walking out the door." Reasons cited for this drain of organizational wisdom included a combination of ongoing economic and competitive pressures on margins as well as the fact that the momentum for outsourcing has resulted in significant (to use jargon of the day) *right-sizing* of the retail banking employee base. Training, for him, has thus become *the* critical mechanism to capture as much of this *walk-around* or *walk-out-the-door* knowledge as possible.

Clark Kelso, CIO of the state of California, in his keynote address at the CIO Academy in California in late 2004, decried the demographic implications of 42 percent of state and local employees as well as over 70 percent of California chief information officers scheduled to retire within the next five years. "What will be the implications on systems maintained and developed," he queried. What will be the implications on the quality of constituent services that these systems enable as these people retire? As Clark pointed out, and a surprise to no one, many of these systems and applications are undocumented, held together merely by *tacit knowledge*—the knowledge in the heads of the people who built them and maintained them over the years.

At a late fall 2003 conference in Europe, the vice-chairman of one of Europe's and Latin America's premiere financial institutions articulated concerns about the "demographic transformation" of Europe and Latin America and its implications for maintaining a leadership position. As the graying (and increasingly, retiring) workforce in Europe meets the labor migration from North Africa and Eastern Europe into Western Europe—and these meet the "demographic greening" of Latin America and its explosive entrance into the workforce—what will be the effect on maintaining a competitive position and effective operations? So much of what people know will be lost as they retire; yet the demands for new products and services will explode in urgency and need. Who will be there to train the new workforce? How, he asked, will we minimize the impact of the *brain drain* as the existing workforce moves on?

There are two specific parts to this *intellectual asset* issue. The first is recognizing the role of *tacit knowledge*—ranging from hard assets (systems, applications, and documented business processes) to soft assets (norms, values, workarounds, and undocumented business processes)—as one of the key intellectual assets of your everyday activities (more on this later). This knowledge is embedded into systems, applications, business processes, and norms of behaviors that together form the organizational glue that keeps the business together on a day-to-day basis. The second part is figuring out the *competitive half-life* of these assets, how valuable they are, and for how long. Which of these assets adds competitive advantage to you and in what areas? How long will those advantages last and what do you do with assets that no longer enhance your competitive positioning?

Geoffrey Moore, author of *Crossing the Chasm* and *Living on the Fault Line*, approaches the half-life question of intellectual assets in terms of *core* and *context*. For Moore, what is *core* to your business are those activities and instantiated assets that add shareholder value and add differentiated value; *context* activities are things that have to get done but add no direct value to your competitive position.[2] The challenge, as Moore summarizes it, is to continually evaluate your core and focus on it, and figure out what to do with your context—whether to stop it, reengineer it, or outsource it. For a quick example, ING Insurance, one of the world's largest and most profitable insurance companies, has been involved recently in developing a new policy administration system. Throughout the project, they have been as aggressively focused on understanding what their intellectual assets were as the specific functionality that had to be built. For each process and application, they have asked what type of competitive value each asset provides, for how long, and what to do about it. They recognized that answers to these questions determine if and how to exploit the asset with significant implications on both their current and future competitive capabilities.

2. *Mobilizing your T-Shirts, Turtlenecks, and Suits—a.k.a., Aligning your (inherently and necessarily) diverse teams:* Most people have encountered the frustrations of conflicting organizational activities, silos, redundant processes, and confusing if not complicated governance policies. The common cry for the need to align metrics is stifled by the operational complexity of how to do so given fundamental differences of priorities and focuses. George Colony, CEO of Forrester Research, an industry research group, humorously characterizes an organization as consisting of three types of people: T-Shirts, Turtlenecks, and Suits. In the next chapter, we reinterpret this somewhat tongue-in-cheek classification into something more tangible and actionable, but for now, T-Shirts are the operations people, Turtlenecks the marketing teams, and Suits the management folks. They may all acknowledge the same set of organizational objectives

and high-level metrics that tell how well they are doing in meeting their objectives. However, each group has its own focus, language, and orientation, making their cultures vastly different, and making each group's approach to accommodating change equally different. It is, consequently, no wonder that organizations so seldom achieve the executional consistency necessary between strategy and operations. The differences in approach, understanding, orientation, and perspectives are so different that it makes any type of executional consistency a pleasant surprise rather than an operational norm.

Facing the Challenge: Bridging the Semantic Disconnect

We have all experienced one if not all of the challenges that result from a lack of executional consistency (if not, please contact us as a case study counter example!). They create tremendous amounts of frustration and costs, both personal and organizational. So, why are these challenges so hard to address? Is there anything today that is significantly different from yesterday to allow us to take on these challenges differently? Is there a common challenge or a root cause that, if redressed, can let us actually get done the stuff that needs to get done? On the surface, their different focuses seem so diffuse as to frustrate any attempts to find common ground among them. But can we not find common ground on which to make sense and take action more effectively? The answer to these questions is the same: A strong yes. There *is* a root cause to the inability to execute quickly, consistently, and effectively, and there are methods, tools, techniques, and standards to address it. And that's what we begin to explain, and explore, in the rest of this chapter and throughout the book.

Underlying the challenge of executional consistency is something we call a *semantic disconnect.* Semantics is defined as the sharing of meanings among different people. Thus, a semantic disconnect occurs when different people take away different understandings of what has to get done to reach an objective. Given such different and disconnected understandings, it is no surprise that what results is an ever-widening *execution gap* that over time merely gets perpetuated, widened, and institutionalized. Sound familiar? Ever experienced this? Each of the challenges discussed in the previous section rests on a semantic disconnect.

Let's take a simple example in the form of an abstracted, but all-too-familiar, business interaction. John, a business operations person, has been working with Mary, a technology person, for several months on an application to assess the credit worthiness of new applicants. John represents his organization, where several hundred of his colleagues hope to use this new application to reduce the time and error rate in their processing of new card applications. Mary represents the software project team creating the application. John and Mary have been working together for the past several months with biweekly updates and occasional "friendly user" demonstrations of portions of the application as it was built.

JOHN: [after signing on to the new credit card assessment application that Mary has just submitted to him to evaluate] This isn't going to work for us.

MARY: [taken aback] John, we've been working on the requirements together for months, and I've been providing you status updates and even access to test out certain features for weeks now.

JOHN: [acknowledging the comment] That's true, but this isn't what we need here. For example, look . . . it doesn't refresh the credit configuration calculator to account for changes in the interest rates.

MARY: [a bit frustrated] But John, we worked through detailed requirements for this application. You even signed off on them, again and again. See? [She points to the requirement documents with John's signature on them.] Here is your signature. This is what you said you wanted!

JOHN: [equally frustrated, voice rising] Well, I may have said that this is what I wanted, but this isn't what I meant. I thought I understood what you were planning to build, but this application isn't what we need!

Do you know anyone who has ever had one of these conversations? Have *you* ever had one of these conversations? Have you ever heard—or said—anything like: "I heard what he/she said, but I have no idea what he/she meant." Or, maybe the question should be: Has anyone *not* heard or said something like this?

John and Mary experienced a semantic disconnect, probably due to the fact that business people and technical people have different vocabularies, so communication between them is, not surprisingly, problematic. Remember the T-Shirts, Turtlenecks, and Suits? Each of them has a specific job to do, and that job has a specific language—a way of communicating, understanding, and consequently acting—associated with it. Using job functions at the far ends of the range, to emphasize the point:

- The CEO (a Suit) uses a vocabulary that includes things like top-line growth, bottom-line profitability, share price, earnings, client or customer retention and satisfaction, and so on.
- The senior vice president of a business unit (also a Suit) uses a vocabulary that includes top-line growth and bottom-line growth for the business unit, and also things like productivity, transactions-per-headcount, utilization of resources (or efficiency of resources), customer-share-of-wallet, and so on.
- The people in marketing (Turtlenecks) use a unique vocabulary, like adoption rate, elasticity, demographic, psychographic, market message, and other terms.
- The people in the trenches processing customer-related things—the (T-Shirts) people who approve insurance claims, provide help desk support, judge the credit-worthiness of a mortgage applicant—use a vocabulary completely focused on their day-to-day job, like *validate claimant eligibility*, or *credit customer's account*.
- The analysts and programmers (also T-Shirts) who create and maintain the applications that support the people in the trenches use a technical vocabulary that includes things like system use case, Enterprise Java Bean (EJB) or XML document.

- The technology operations people who install and maintain the computing infrastructure that the applications run on use a unique infrastructure vocabulary that includes things like router blade, storage area network, or SONET ring.

Yet, and more importantly, even if T-Shirts, Turtlenecks, and Suits happen to use the same language, they often mean strikingly different things by the language used. For example, think of the word "account" or "customer." These terms often mean widely different things to different parts of an organization. Ask yourself: How many hours and dollars are spent on clarifying what these terms mean, in getting the head of the business, the IT manager, and the marketing analyst to agree on something that on the surface seems so obvious? But is it? Is an "account" or "customer" someone who purchases one of your products, receives specific services, a data structure in a database, a demographic to be mailed to, a prospect to be sold to, a statistic to be measured, an asset to be exploited, a cost to be handled, an opportunity to be positioned to, a risk to be managed, or one of a dozen other descriptions?

Which characterization is "best?" Which is most appropriate? The challenge is that each of these characterizations might be appropriate for what has to get done by the person who has to get it done. Moreover, there is seldom a way to show how these different understandings connect with each other—and impact each other. It is this connectivity that is critical to resolve the semantic disconnect across these different relevant and equally important understandings of what constitutes an account, or a customer. Creating such connectivity—demonstrating how they impact each other and thereby make sense to each other—is what we need to do to get the alignment desired and the executional consistency needed.

Okay, you say, what's different now, and how do we make it happen? Let's answer those questions in two parts: first, how to think about it, and second, how to do it—how do we make it real.

In day-to-day operations, when things are working well and the company is conducting business as usual these groups don't have to interact much. Where they do interact in the course of day-to-day business, they have worked out common semantics over time and much effort.

During times of change, however, semantic differences among different groups of people become barriers to executing consistently to make the change occur. In our earlier example, the change was that John, a member of the population in the trenches with the customer, wanted a change—specifically a new computer application from Mary, a member of the analysts and programmers population. John and his team articulated their needs in terms of their group's semantics. Mary and her team translated those needs in terms of theirs and created a computer application. We saw the result: The application initially missed the mark. Over time, John and Mary will go through a few more iterations of the application and it will perform as John and his team envisioned. However, the extra iterations add time and cost to the process and, depending on how dynamic John's part of the business is and how long it takes to get the application right, the business needs for the application may have changed by the time the application is ready, setting off another round of changes. (Not that this has ever happened in your business, of course.)

Over time, John and Mary will themselves create shared semantics; as a team of "businessperson" and "tech person" they will become much better at getting specifications and the resulting applications right the first time. They will have bridged, or at least narrowed, the semantic disconnect between them. The benefits will be substantial, but will last only as long as John and Mary are the two people working together. As soon as one or both move on to new job assignments, the semantic disconnect will open wide again, and the attendant inefficiencies will reemerge.

The same disconnect issue arises, on an even more expansive and expensive scale, when the CEO decides on a change in corporate direction. In their book *Execution: The Discipline of Getting Things Done*, Larry Bossidy and Ram Charan focus on the need for the CEO to be totally immersed in the organization and to drive the linkage among strategy, people, and operations. Ram Charan has said that "if you have a five-year vision that's good and inspiring [and] you'd better convert that vision into a two-year vision. If you don't do that, you lack the skills to be able to be an effective leader."[3]

Execution focuses on the disciplines of selecting and evaluating the right people to drive the organization, on the importance of setting goals, following up to be sure those goals are understood, measuring progress toward those goals, and rewarding people accordingly. Bossidy's application of these principles at Allied Signal—leading to 31 consecutive quarters of growth in Allied's earnings per share of 13 percent or more—speaks directly to the power of consistent execution. Our approach marries some of the ideas and practices in *Execution* with the powerful concept of *closing the semantic disconnect to drive execution pervasively through every business unit, department, and ultimately every person in a company. Execution* illustrates the potential of aligning the leadership of an organization and making each layer of leadership responsible and accountable for the performance of its part of the organization. However, even at the height of its performance, interactions like those between John and Mary were happening throughout Allied Signal every day, not due to a failure of leadership, but because workers spend most of their time and energy on doing their assigned job (their "day job"), and that job has its own demands, jargon, and semantics.[4]

Good leadership includes setting goals, setting schedules, following up, measuring results, rewarding and coaching the people involved. Yet, leaders seldom know—in detail—each interaction required to complete a set goal. There could be a smooth interaction between people who know each other well and understand a shared problem. There could be an arduous, uphill "death march" with enormous struggle and waste, absorbed in heroic efforts on the part of the people involved, late nights and weekend work, with the team meeting its goal but becoming bruised, burnt out, and disillusioned along the way. As detailed in *The Jericho Principle*, one major contributor to the difference between a smooth result or a brutal death march is the presence or absence of shared semantics among the participants.

Semantic disconnects occur again and again in organizations all over the world, leading to extra work, additional costs, finger pointing, personal and personnel frustrations, budget blowouts, exacerbated organizational silos, and (need we add) project delays that sap the organization's capability to mobilize in response to a need for change.

Now, assuming the semantic disconnect *is* the root cause hindering executional consistency, the simple follow-up question becomes *how* to semantically *re*connect. *Consequently, the critical requirement to drive executional consistency is to overcome the semantic disconnects that permeate many of your organizational activities.* This means having the tools and methods to create common expressions to drive the executional consistency needed. It is fundamentally about creating the capabilities to ensure that strategic decisions taking place at the top of an organization are consistently executed down and throughout an organization—through its business processes, applications, and infrastructure. As we said earlier, this is not about everyone knowing everything about all parts of the business; that is neither feasible nor desirable. What *is* possible, however, is to use tools and methods to let each organization use its natural vocabulary, while at the same time understand how the semantics of each part of the organization relate to every other part of the organization. With what result? Visibility and connectivity among the different parts of the organization that help establish common understanding, align metrics, rationalize governance, streamline communication, and thereby create the capabilities for consistent execution.

This is a nice vision, but one so hard to achieve that it has become in many people's minds the business equivalent of King Arthur's search for the Holy Grail. But it needn't be: There are tools, methods, technologies, and standards that exist now, some that have been around for a while, some that have begun to mature within the past year or so. We'll explore some of these later. But first we need to describe how to think about—make sense—and take action on creating the shared semantics so sorely lacking, but so critically needed.

Answering the Question of How

So far, we've described core challenges to getting stuff done consistently, described requirements to getting it done, provided some quick examples, and presented the underlying DNA *for* executional consistency. But we have yet to describe how to look for that DNA and how to manipulate it for your purposes.

Getting stuff done consistently entails two activities. First, it requires that many people rapidly *make sense of*—understand—what it is that has to get done. Second, they must effectively *take action* by mobilizing an array of people and resources *to* execute consistently. Let's take a deeper look at what it takes for this to happen.

Change in an organization happens when someone has an idea, recognizes an opportunity, or responds to a threat that others haven't identified. The insight initially resides solely in its creator's head—in other words, as *tacit* knowledge. Taking effective and scalable action depends on taking this tacit idea and mobilizing that idea so others—ideally, many others—can understand, then execute that insight on a large scale. This capability to take action on a large scale requires *codifying* the tacit idea—turning it into an explicit form that then can be communicated, understood, and used by many others.

To illustrate this codification process, think of a chef's signature dish served at a restaurant. As long as the knowledge of how to create the dish remains the tacit knowledge of the master chef, he or she can create the dish for a handful of diners. If, however, the knowledge of how to create the dish is codified into a recipe, the dish can be created in many places by many chefs, and can become a global franchise. *That's scale, and the secret sauce is codification, the recipe to everything from potato chips to microchips.*

Tacit knowledge is one of the key intellectual assets of your everyday activities (more on this later). *Executional consistency speaks to the process of codifying tacit knowledge.* As tacit knowledge is codified, it moves from being locked away—in places like the heads of critical individuals, in software logic, or in application databases. It is captured in and usable through a growing repository, using well-defined expressions of business processes, technology enablers, and other assets. Such expressions often take the forms of models, equations, simulations, scenarios, and other forms of documentation that create the shared semantics that everyone in the organization can use to execute—consistently, effectively, and cost-effectively. (We often characterize these and other types of expressions as "grammar tools"—for a simple reason. Grammar is the inherent building blocks of effective communication, in turn necessary to have a hope of shared understanding and consequently effective action. This is why when we often hear in discussions, "oh, that's just semantics," as people attempt to dismiss-away disagreements or understandings, we quickly pounce and reply "but semantics are all we have" and why means to express them, and act *from them*, are so critical.) Tacit knowledge (*all those intangibles of what we know and experience*) itself has potentially high value, but, being inaccessible, is not scalable. *We break through the scaling limits of tacit knowledge by codifying that knowledge into the processes, frameworks, and standards where the power of scale can kick in.*

How well a company does this is the true determinant of how successfully it can execute consistently and at scale. We've reduced this claim to a simple mantra: *the more codified, the more executable; the more executable, the more scalable; the more scalable, the more consistent, aligned, and able to change rapidly, effectively, and efficiently.*

The Semantic Stack as Insight and Action

What do we mean by codification and how do we know it when we see it? One conceptual tool we use is called the *semantic stack*, a simple way of thinking about the degree of codification and its impact on what you do on a daily basis. But before describing the semantic stack, let's step back and explore the question: What do we mean by codification and how do you know it when you see it?[5]

At some point in the distant past, people began expressing themselves using words, first spoken and then written. From these words emerged common ways of speaking, shared communications, common viewpoints, shared culture, and the capability to mobilize people to effective action. So, what's different now? Not a thing! Organizations consist of people with different personalities and perspectives, different departments and duties, different ways of thinking and acting. What made and makes effective actions are the *shared understanding of what is expected, of what to do, of how to do it, and of measuring and realizing the value of those actions.* Shared semantics creates this shared understanding and enables scalable action. Why is this important? Because shared knowledge and understanding becomes more scalable and cheaper when knowledge is codified. Consequently, the degree of codification in any area is a critical measure of organizational agility, and of being able to execute consistently across the T-Shirts, Turtlenecks, and Suits. Now, back to the semantic stack.

In Figure 1.1, we show the semantic stack as a simple grid. The vertical dimension of the stack refers to different types of organizational activities. Roughly speaking, as you move from the top of the stack to the bottom of the stack, you are moving from overall corporate strategic things, through business operations, and down to the technology that supports the business. Imagine a senior executive who wants to make a change in corporate direction: that change at the top of the stack can only be realized by changes through the layers lower in the stack. Similarly, imagine a fundamental change in connectivity infrastructure, such as that resulting from the Internet. Such a change can result in change rippling upward through the stack, ultimately affecting the strategic direction of the business.

Philippe is executive vice president of Global Cash & Trade—one of the world's largest banks. His decision to expand aggressively throughout Eastern Europe is a strategic decision to take advantage of ever-shifting regulatory changes allowing financial capital to flow more freely throughout the region. Making this decision is one thing; acting on it requires concentrated efforts and aligned capabilities from his strategic intent through business process coordination, through underlying application extensions, and through network and infrastructure deployments. As we have all likely experienced, getting the

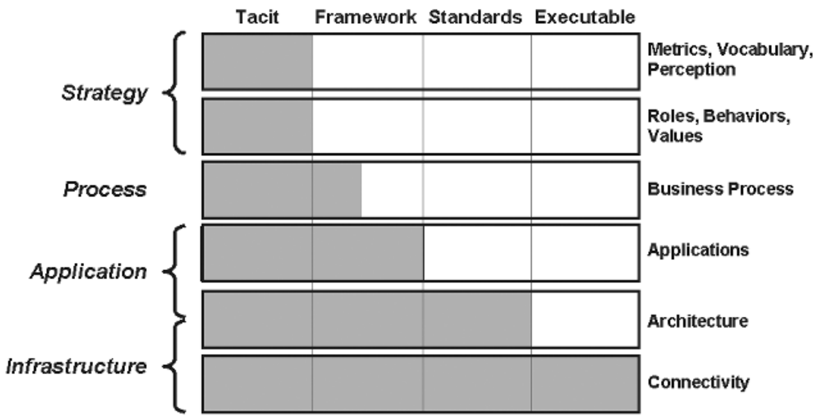

FIGURE 1.1 **The Semantic Stack**

alignment in terms of mobilizing activities within and across these different areas is difficult. Another difficulty lies in knowing how they connect and (will) impact each other. Philippe's decision is a "top-down" one starting from strategic intent impacting "down" into his infrastructure. But such impacts, dependencies, and constraints work the other way as well—rippling up through the stack, ultimately impacting the strategic direction of the business. Have you ever heard, or said, "we can't get that new product out the door because our network, or applications (or some other IT areas) cannot support it"? We talk much more later about this challenge and how to tackle it based on creating executional semantics.

The horizontal dimension of the stack represents the degree to which the knowledge of the layer has been codified. The further to the right, the greater the codification. The shading in Figure 1.1 gives a rough estimate of the current state of codification in each of the layers. This indicates, for example, that industry standards bodies, vendors, and the majority of global businesses have fully agreed to the standards and *shared semantics* underlying infrastructural connectivity around IP—the communications protocol that enables the Internet. Before such shared and executable semantics were created, the industry had a Babel-like environment of competing and conflicting ways of plugging into networks. Now there's a "standard" network, agreed to by most people, enabling a degree of automation and automatic execution that far exceeds any other set of organizational activity in this framework.

The upper layers of the stack remain less codified because, frankly, there is little or no agreement, within companies much less across industries, on a common way to represent activity and knowledge at those layers. Take the business process layer, as an example. We started this chapter talking about Mike's challenge of being sufficiently responsive to the market with his 150 plus retail banking products and underlying applications. The processes that Mike's technology needed to support were created over time, by different people, meeting different market needs. Mike needed a way to communicate about these processes—to express them—in a way that the business, the marketing, *and* the technical people could understand. He created business process models because without them there wasn't a consistent way the T-Shirts, the Turtlenecks, and the Suits could talk about—much less understand—what made up the business. And, without being able to do that, there was no way to be able to change the business quickly. But interestingly, while Mike recognized the importance of codifying the bank's processes, there is no industry-wide standard of doing so. Using the semantic stack model, Mike began codifying the tacit knowledge of the thousands of people at his bank and the work they did to support their customers in a way that could be meaningful and provide significant advantage to the bank. However, when Mike's bank acquires yet another bank—which it does on a frequent basis—the integration of the existing processes with the new processes will still be a challenge because there remains no cross-bank or cross-industry shared semantics of how to model processes. This is why the business process layer of Figure 1.1 is shaded.[6]

Let's compare the shading in the connectivity layer with the business process layer: while the Internet protocol specifies clearly how information flows on the network, there is no correspondingly universal way to describe business process flows, such as the process by which a credit card company takes in a customer request for a card and ends up issuing a card and opening an account or denying the card and reporting it to the credit bureaus.

The lack of codification in the upper layers of the stack means that people within those layers have higher barriers to accommodating change: When change in a layer is required, and the level of codification in that layer is low, then the people who must execute the change must first come to a semantic agreement—must close any potential semantic disconnects—about what the change is and what it means. Until that happens, many "John and Mary" interactions will *slow the process of change, and introduce ambiguity, which takes away the ability to execute consistently, effectively, and efficiently.*

Therefore, in terms of the stack, moving from *tacit* to *executable* across more and more of the layers of the stack increases the potential for executional consistency. We call this process *walking up and across the stack.*[7]

We can use the semantic stack and this process of *walking up and across the stack* to provide insight into competitive dynamics in general.

Competition results from different companies attempting to exploit a sufficiently attractive market opportunity. Initial market opportunities are usually high-margin and/or high-revenue opportunities, the results of their underlying value propositions being novel, consequently relatively unexploited or difficult to replicate, and largely embedded in the heads of relatively few people. Over time, these margins tend to get arbitraged away or shrunk as new competitors, recognizing the potential of those market opportunities, enter the competitive fray. What shrinks those margins are processes, technologies, and other activities that bring down their operational costs and allows them to become more scalable, hence executable by many. The means of driving such scalable activities is the enabling codification of those activities—of the tacit knowledge, the knowledge in the heads of few—into frameworks, into standards, into executable and repeatable activities.

So what? For now, let's highlight two things of importance. First, the stack serves as a mirror you can hold up to your business and see what is reflected back. By showing the degree of codification in a particular layer, the stack gives you a means—in an abstract but provocative way—to see how your company measures up. Again, back to Mike. Mike recognized that he had to begin to codify his processes; the bank is growing too much, is too global with too many different constituents and customers not to have widely shared agreement of what the business does, and how Mike's organization fits. "We're global . . . with the potential of fragmenting into dozens of competing companies. . . . We can't afford to do that; the risk is too great." Attempting, aggressively, to create shared semantics of what applications supported which products where, when, and how was a competitive necessity for Mike to become more effective at what the bank had to get done.

Second, the stack shows how the intellectual assets in your company are stratified, from abstract things like "business objectives" to concrete things like "connectivity" and "platforms." This is important because executional consistency comes from knowing what connects to what, when, where, how, and how much—or using the more formal language we've introduced here, from codification within a layer and codification in how adjacent layers connect with, and impact, one another. To make this point, let's focus on two of the layers of the stack—the applications layer and the business process layer—between which some of the worst semantic disconnects occur. We discuss codification within each layer, then codification between the two layers. This provides the foundation for the material in Chapter 2 where we detail Business Blueprinting—a specific set of methods, techniques, and disciplines to codify tacit or simply widely disparate knowledge both within and across the layers—across the very different organizational activities and perspectives of your T-Shirts, Turtlenecks, and Suits.

Detailing the Stack

The business process layer is where the company's production happens and where the company interfaces with the outside world, for example, its customers and its partners. A business process consists of a set of activities to get something done; processes describe the functions and interactions among people, departments, and companies defining the rules and procedures by which the company runs as well as the ownership and flow of information to get that something done. Codification in the business process layer means having clear expressions of job responsibilities, workflow, decision points, information flow, and business rules. The semantic stack indicates that the business process layer is not as codified as the application layer. One big reason for this is that there is no commonly accepted discipline for describing much less modeling, analyzing, and engineering good business process. Nor is there common agreement on even how to express "a good" business process design.

Instead, in many companies, organizational processes are documented in practices that no one actually reads. Knowledge of the way processes work is passed down as "organizational lore" over time or depends critically on the knowledge in the head of some key person(s). Executional consistency demands that business processes rapidly and precisely change as the needs of the business change, and that different parts of the organization that are doing essentially the same thing do it the same way, hopefully using common supporting technology. This requires that, like the recipe discussed earlier in this chapter, business process is recorded, understood, and accessible to everyone with a need to know. Again, such leverage was precisely Mike's motivation to do the work we have been discussing throughout this chapter.

Business process codification requires a commonly accepted way or a standard of expressing processes. Emerging languages based on eXtended Markup Language (XML), such as Business Process Execution Language (BPEL) and Business Process Modeling Language (BPML), provide a means to enable task sharing within and across organizations. Given that there is more than one way to express business processes, there remains codification to be done. Clearly though, businesses seeking agility need to begin the process of describing and defining business processes in terms that can be readily expressed, and hence understood and executed (pun intended) on to different parties in whatever standard emerges. (Skip to the next section, "Making It Real," if you want to be saved from a fairly technical discussion.)

The applications layer is where automation supports business process. Today that generally means computer applications, but can also mean physical machinery like a check sorting machine. In the applications layer, referring now to computer applications, codification translates to, among other things, clearly delineated application functionality and clearly specified interfaces between applications. Taking them one at a time:

- *Application functionality:* Applications exist to support business processes. For most of the time since people have been writing computer applications, they have focused on creating them to support one or a few linked business processes. The problem with that approach is that, in rapidly changing times, business processes need to change much faster than applications can be reengineered. For example, a new wireless telephone carrier can offer calling plans with billing in one-second increments with rollover. Legacy carriers have been challenged to respond rapidly to this competition partly because their billing applications have a legacy back to times when the telephone industry was stable (and people were billed "for the first three minutes"), and the billing applications did more than just billing for phone use—they included functionality for marketing, fraud detection, tariffs, and many things not strictly related to billing.

More recently, application development disciplines such as Object Oriented Design (OOD) have evolved that allow software engineers to understand problems in terms of clearly partitioned services. Applications created using these new engineering principles comprise discrete components, each of which implements a self-contained service that can be extended or modified with minimal impact on other components. Thus, a well-designed biller created using OOD principles can be easily modified to accommodate billing increments other than "the first three minutes." OOD and related disciplines provide codification that creates the conditions for scalable understanding and execution.

• *Interfaces:* Applications need to communicate with other applications, both within an organization and among organizations. Again using the telephone company example, opening a new customer account requires that the biller communicate with the customer care applications, with the provisioning applications that turn on new service, and with the billers of other telephone carriers. For applications to interface, there needs to be an open channel and a clear language. Codification in interfaces means widespread agreement on the communications channel and on the language "spoken" across that channel. At one time, interfaces were created one application at a time with slight regard for other application interfaces and thus, little standardization.

Over the past few years, however, the Internet and its related technologies have emerged as the clear channel for communications between applications and organizations. Along with this, languages like HyperText Markup Language (HTML) and eXtensible Markup Language (XML) have emerged as standard languages for application-to-application communications. Specific vocabularies of XML such as e-business XML (EBXML) and security assertion markup language (SAML) provide widely understood and accepted means for communicating among applications. Finally, Web Services, an emerging standard for application-to-application interaction across the Internet, provides a standard way for applications to advertise and fund functionality, connect, and communicate via XML.

Interface codification creates the conditions for agility since applications (supporting business function) can more easily and robustly communicate with other applications, meaning that as business functions change, applications can be rapidly deployed to support the change. Codification within the application layer and the business process layer allows each layer to be engineered and managed more efficiently. We stated earlier that executional consistency depends also on codification between the layers. Most business processes are supported in some way by application functionality, and conversely, most application functionality exists in support of some business processes. Thus codification between the business process and application layers means that there exists clear *traceability* between a business process and its supporting application functionality. Traceability means that the impact of a change to a business process—to support a new business opportunity for example—can quickly be translated into necessary changes to underlying applications, speeding implementations and increasing agility. Similarly, traceability means that changes in the application layer—to accommodate a new release of vendor software for example—can be evaluated quickly and accurately for possible impact to business process, eliminating nasty surprises.[8]

Making It Real: Executing Over and Over Again

Our composite made-up examples of John and Mary, the real-life ones of Mike and Clark, and possibly many of yours reflect what many of us know too well: Doing the stuff that aligns your firm's objectives with everyday activities is hard; execution is hard work, and consistent execution even harder. And, of course, there are many examples and excuses, situations and rationales for why disconnects exist both within and among the T-Shirts, the Turtlenecks, and the Suits. Yet, underlying them all, underlying the challenges to getting stuff done consistently, lies a simple but insidious disconnect: the semantic disconnect.

Here's where it gets interesting and useful. If we understand that the semantic disconnect is the root cause impeding executional consistency, then a new way of thinking, of pragmatically acting, becomes clear: *Develop the tools and methods to bridge the disconnect, creating execu-*

tional semantics that are as meaningful as they are actionable by everyone. This is not a group-think thing. Rather, it is exposing the proverbial "elephant on the table." Group-think cannot connect the semantic disconnect. Diversity of perspectives, personalities, and actions is far too critical to keep insight and innovation, excitement, and opportunities for those organizations who nurture them thriving and highly competitive. Besides, the different audiences—executives, business process owners, application writers, and so on—really do need to have their own distinct vocabularies, and understandings, for an obviously simple reason: Their jobs *are* different. Yet, respecting differences is one thing; respecting them and knowing how to marshal them in a consistent, effective, and pragmatic way another. And it is the latter we're focused on.

In the last scenes of the movie *Amadeus*, Salieri is shown scribing the music of an ill Mozart, performing the role of an amanuensis—a person who is translating the brilliant insights of Mozart into a form that can be read, interpreted, and performed by thousands if not millions of people. He is *codifying* Mozart's *tacit knowledge*—creating the platform to scale, to execute, over and over again. And that is the simple but powerful key to the semantic stack, which we will bring operationally alive with Business Blueprinting in the next chapter. Bridging the semantic disconnect entails creating tools and methods that allow the T-Shirts, the Turtlenecks, and the Suits *with their own perspectives and behaviors* to work together—based on creating a shared environment that is as meaningful to each of them as it is actionable by them all. And codifying the tacit knowledge that *is* the expression of all of these differences *is* the DNA for executional consistency and the strands to manipulate to get done what needs to get done—quickly, consistently, effectively.

What This Means to You

1. *The semantic stack is pretty abstract; what use is it to me?* We've spent this time explaining the semantic stack and the underlying concepts of codification and the semantic disconnect for two reasons: First, to provide the framework underpinning the rest of this book—the

semantic stack is the conceptual cousin of the very concrete Business Blueprint we introduce in the next chapter.

Second, to act as a "mirror" with which you can view your own organization. The semantic stack is a qualitative way for you to look at the activities in your organization and ask yourself key questions, such as:

- What are the key reasons I'm challenged to get stuff done?
- Is there something, some set of reasons, underlying or common to these challenges?
- How well documented and understood are my business processes?
- How dependent am I on a few key individuals, applications, and/or systems who hold the knowledge on how things work?
- In meetings among departments in different layers of the stack, how much time is spent wrangling over the meanings of terms and objectives? How often do people use the same term (e.g., "customer" and "account") to mean different things? How often do people use different terms when referring to the same thing?

Based on asking these sorts of questions, you can judge where you are and how far you need to go in applying the methods, tools, and techniques that we detail in the next chapters.

2. *Are you saying that everyone across the organization needs to be connected to everyone else?* No. There is a common misconception that there's no such thing as too much communication. We disagree completely. Over-communication wastes time and resources and is a poor substitute for the *right* communication. Executional consistency depends on creating the shared vocabulary and the shared semantics so that you know what communication is necessary where, when and how, and so that the necessary communication can take place with minimal effort and loss of understanding. Again, one of the challenges we see over and over again is that within "communication," there is lots said, but not much heard.

To be effective, communication needs to be understood, and actionable. But to be understood and actionable requires "talking in the language"—or delivering the content—in a way *that is as meaningful as it is actionable to whomever needs it, and critically, how they can use it.* Because the T-Shirts, Turtlenecks, and Suits all have different needs, means of understanding, communicating, and acting, creating "shared semantics" is not an issue merely of "communicating" or "communicating more;" it is one of creating a shared semantics so understanding, and pragmatic, execution can occur. This is why we focus so much on the models, the messages, the software, the business rules, the (un)documented business processes—and all the other means of "expressing" what is done within an organization—codifying what is in someone's heads so others can use it.

3. *So, "codification" is pretty key?* Absolutely. Codifying the intellectual assets in your organization to bridge the semantic gaps that naturally exist in any organization *is crucial.* How well a company does this determines how successfully it can execute consistently and at scale. We've reduced this observation to a simple mantra: *The more codified, the more executable; the more executable, the more scalable; the more scalable, the more consistent, aligned, and able to change quickly, consistently, and effectively.*

Let's expand a bit on this answer. A business's intellectual assets include knowledge in people's heads, logic buried in computer programs and databases, information in a knowledge management or business intelligence portal, and/or the processes—documented and undocumented—that you use to run your business. Codifying those assets means expressing them in a way that is (1) understandable and accessible to everyone that needs to use them directly; and (2) connected in a visible, traceable way to the knowledge and assets used by other people in the organization. The semantic stack provides a way to discuss codification, with each layer of the stack representing a distinct function of an organization that can benefit from codification (the "everyone that needs to use it" mentioned earlier); the stack in total characterizes organizations as a bunch of smaller organizations that have to work together (the reason there needs to be "a visible, traceable way" that the knowledge of one organization relates to another).

We refer to codification as the "DNA for executional consistency." The semantic stack is to executional consistency what the double-helix model is to biological DNA. Both give us a mental picture with which to begin to understand a complex and important system, and expose some of its underlying constituent elements. Neither gives a prescription for how to manipulate those constituent elements effectively to create a desired response. Much of the remainder of this book is devoted to "unpacking" the semantic stack into a set of very tangible things you can do to increase codification and thus vastly improve how to get done what you need to do to enhance the performance of your organization. Specifically, in Chapter 2, we examine in detail methods, tools, and processes that can be used to create those characteristics, and in Chapter 3 we explore some methods and processes to actually change—permanently—the organization to realize that *executional consistency thing.*

4. *This seems huge. Is there a way to get started, and do it piecemeal?* Yes. It has to be done piecemeal because it's impossible to do it any other way, and we will show how in the next few chapters. Once a part of the organization has internalized the discipline to make codification a priority outcome of any project, then every project completed increases your capability to be more executionally consistent. This means that you can start pretty much anywhere. A good way to start is to examine your top 10 strategic initiatives, "grade and shade them" against the semantic stack in terms of the degree to which the project activities are codified, and then programmatically begin with codifying the patterns you see across these top 10 projects.

5. *My organization changes constantly because of competitive pressures, changes in regulations, and changes in customer demands . . . when can I find time to "codify" things?* We would ask in response, "How can you afford *not* to find the time to codify things?" As codification increases, executional consistency increases and you can respond more rapidly, effectively, and efficiently to change. And, as hinted at in the answer to the previous question, the discipline of codifying a project at a time makes for better projects, and actually adds little or no cost. The disciplines you put in place will reduce the execution risk of each project on its own, helping you to change with predictable success. We'll see multiple examples of this throughout the rest of the book.

6. *Are there technologies and standards that can help me?* Yes, to varying degrees in different parts of the stack. We will examine this in more detail in the next chapter.

7. *If I start doing things in "standard" ways, how do I keep my differentiation from my competitors?* When we preach codification and standardization, we are talking about codifying the way *you create* your differentiation, and exploiting standards so that you can more rapidly, effectively, and efficiently create and deliver the products and services that differentiate you in the market.

8. 4. *How should leadership interact with the different layers of an organization?* Let's quote Ram Charan again regarding the plight of technology organizations as an example of one layer in the semantic stack: "IT people are forced to set their own priorities. They love to have the involvement of senior leadership. Those who have it, they are succeeding; those who do not have it, have been left behind."[9] Executional consistency means creating the capability to mobilize leadership thinking into the decision-making process of every part of the organization. Business Blueprinting lets everyone in an organization approach the solution to a business problem in a coordinated, coherent way; each sees and contributes to the parts of the problem that they can best help with, and the Blueprint connects them to the contribution of everyone else involved in solving the problem. We'll see this in detail in the next chapter.

Chapter Cheat Sheet

The Issue

Many businesses face a nearly insurmountable challenge in getting done what needs to get done, in aligning their goals with their execution. No doubt, such alignment is hard to get, and the resulting challenges to do so are significant. But why? What is it about this "executional issue" that makes it so difficult? Lots of answers are offered and recommendations made to these questions. Still, challenges continue to occur in getting the executional consistency so sorely needed but so woefully lacking. But they need not. Rather than getting bogged down in the myriad of possible explanations of why such alignment doesn't occur, let's start with a simple, but provocative question: Is there a common challenge or root cause underlying these challenges that, if understood, could serve as the starting point to overcome them and thereby build our capabilities to get the stuff done that needs to get done? The answer is yes. There *is* a root cause to the inability to execute quickly, consistently, and effectively. Understanding this underlying root cause, *what we call the DNA of executional consistency*, is critical to being able to manipulate it.

The Insight

Underlying the challenge of executional consistency is something we call a *semantic disconnect.* Semantics is defined as the sharing of meanings among different people. A semantic disconnect occurs when different people take away different understandings of what has to get done to reach an objective. In day-to-day operations, when things are working well and the company is conducting business as usual, groups don't have to interact much, and where they do interact in the course of day-to-day business, they have worked out common semantics over time and much effort. During times of change, however, semantic differences among different groups of people become barriers to executing consistently to make the change occur. Given such different and disconnected understandings, it is no surprise that what results is an ever-widening *execution gap* that over time merely gets perpetuated, widened and institutionalized.

The Phrases

Semantic Disconnect; Semantic Stack; Tacit Knowledge; Codification; T-Shirts, Turtlenecks, and Suits; What Connects with What, When, Where, How, and How Much

The Implications

If we understand that the *semantic disconnect* is the root cause impeding executional consistency, then the next step becomes clear: *develop tools and methods to bridge the disconnect*, *creating "shared semantics" that are as meaningful as they are actionable by everyone.* Because the T-Shirts, Turtlenecks, and Suits all have different needs, means of understanding, communicating, and acting, creating shared semantics is not an issue merely of communicating or communicating more, it is one of creating a shared semantics so understanding, and pragmatic, execution can occur. This is why we focus so much on the models, the messages, the software, the business rules, the (un)documented business processes—and all the other means of expressing what is done within an organization—codifying what is in someone's heads so others can use it. This is not an issue of changing your languages, of aligning your metrics, of changing your governance processes, of communicating more, and more. It is fundamentally about creating shared and shareable capabilities to drive consistent execution. It is fundamentally about creating capabilities to ensure that decisions taking place at the top of an organization can be and are consistently executed throughout the organization—through its business processes, applications, and infrastructure.