Praise

This book should be required reading for all technology and business leaders who are serious about digital transformation. It takes you on a provocative, fun, and comprehensive tour of the key areas that will promote and ignite agility, creativity, learning, community, and collaboration.

This book may be about taking a seat, but this is no time to be sitting still! IT leaders will be convinced that their job is now about incentivizing and inspiring courage, passion, and technical excellence in service of business objectives rather than blindly servicing requirements. You will find practical advice on how to deal with projects, scope creep, IT assets, governance, security, risk management, quality, and shadow IT.

—**Jason Cox, Director**, Systems Engineering,
The Walt Disney Company

In his first book, *The Art of Business Value*, Mark brought together a unique understanding of modern techniques—Agile, DevOps, and Continuous Delivery. In *A Seat at the Table* he grabs hold of these concepts and disrupts the conventional dynamics around the role of the CIO in any organization. His progressive thinking is unmatched and a must read for leadership and practitioners of all kinds.

—**Luke McCormack**, former CIO of the
Department of Homeland Security

Mark has found the IT leadership cheese after Agile moved it. Finally, an idea of how to structure IT, including leadership and the teams, and joining the business and IT together!

—**Joshua Seckel**, Chief Engineer at WhiteHawk CEC, Inc.

High-performing organizations see technology as a strategic capability of their business. The walls, inertia, and confusion of seats, sides, and responsibilities does not exist for them. Yet many organizations still retain legacy mind-sets and behaviors that limit their opportunities to improve, innovate, and inspire their people. Mark shows the steps needed to break free of these challenges and unlock potential, speed, and growth. His advice is pragmatic, practical, and to the point.

—**Barry O'Reilly**, co-author of *The Lean Enterprise*

"Agile" is more than a new software development practice; it is a new way to think, engage, and lead. As Mark Schwartz points out in his compelling new book, *A Seat at the Table*, when CIOs re-conceptualize their role based on Agile principles, they will stop worrying about having a seat at the table and start realizing all of the full potential of IT.

—**Martha Heller**, CEO of Heller Search Associates and
author of *Be the Business: CIOs in the New Era of IT*

I use to feel guilty when someone would ask me how do I get my leadership to understand DevOps if they refuse to accept it. My answer was, basically, you can't. Now I can give them a copy of *A Seat at the Table*.

—**John Willis**, Co-Author *The DevOps Handbook*

Fresh thinking and useful advice fill the pages of Mark Schwartz's *A Seat at the Table*, which strikes an encouraging, instructive tone about the future of IT leadership and the CIO's expanding business role. "If we cannot know the future, then we have to think a bit differently," he writes. And he does just that. Mark's argument that IT executives must change their behaviors—dropping the "command and control" mindset in favor of community building and Agile leadership practices—resonates throughout this well-organized, thoughtful book. While attaining that "seat at the table" often refers to CIO career goals, the ideas and approaches explored in this book are essential reading for anyone hoping to advance in the IT profession today.

—**Maryfran Johnson**, Executive Director of CIO Programs,
IDG (International Data Group)

Mark Schwartz is a rare combination: a deep thinker who has also applied lean, Agile, and DevOps principles at the highest level, leading an extraordinary Agile transformation in the US Federal Government at USCIS. In this book, he shows how modern IT leaders succeed by driving business outcomes rather than operating an order-taking function. This shift in organizational mindset is critical to any successful technology transformation but requires substantial changes in behavior at every level, and Mark's thorough analysis will prove invaluable to leaders who must execute it.

—**Jez Humble**, CTO, DevOps Research & Assessment LLC

If you're a CIO, read this book. If you're not a CIO but work closely with one, read this book. Mark Schwartz is the best of iconoclasts. He brings deep insights from his unique erudition and real-world experience—ranging from a startup to government agency—in untangling the dilemma of the CIO in the second decade of Agile. There aren't many people who can swing from Horace to Daniel Pink without losing a breath. And there aren't many who can critique Agile and Waterfall with equal insight. This is a surprising book—well worth your (20%) time.

—**Sam Guckenheimer**, Product Owner,
Visual Studio Team Services, Microsoft

As with his book *The Art of Business Value*, Mark Schwartz directly confronts the tensions that exist across the corporate IT landscape, showing us how we got here and what to do about it. Almost every page contains a situation I've seen in my day-to-day work, but that have not been articulated before. [*A Seat at the Table* is] required reading for anyone seeking to understand how IT should work with an organization to achieve success in an Agile age.

—**Ian Miell**, Lead Software Architect, Financial Services

a Seat *at the* Table

IT REVOLUTION
25 NW 23rd Pl, Suite 6314
Portland, OR 97210

First Edition
Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

Cover and book design by Devon Smith
Author photograph by Gary Landsman

Library of Congress Catalog-in-Publication Data
2017948982
ISBN: 978-194278-8119
ePub: 978-194278-8126
Kindle: 978-194278-8133
PDF: 978-194278-8140

Publisher's note to readers: Many of the ideas, quotations, and paraphrases attributed to different thinkers and industry leaders herein are excerpted from informal conversations, correspondence, interviews, conference round-tables, and other forms of oral communication that took place over the development and writing of this book. Although the author(s) and publisher have made every effort to ensure that the information in this book was correct at press time, the author and publisher do not assume and hereby disclaim any liability to any party for any loss, damage, or disruption caused by errors or omissions, whether such errors or omissions result from negligence, accident, or any other cause.

For information about special discounts for bulk purchases or for information on booking authors for an event, please visit our website at ITRevolution.com.

a Seat *at the* Table

MARK SCHWARTZ

IT Revolution
Portland, Oregon

To the talented and hard-working government employees,
so resilient in the face of impediments, criticism, and abuse.
I have so much fun working alongside you.

CONTENTS

INTRODUCTION

The demand that I make of my reader is that he should devote his whole life to reading my works.

—James Joyce, *interview*

Let these things be believed as resembling the truth.

—Xenophanes, *Fragments*

As I was writing my last book, *The Art of Business Value*, something at the back of my mind kept nagging me. It finally made its way onto the page in a chapter I wrote called "The CIO," where I looked at the role of the CIO in delivering and defining business value. The issue, as I saw it at the time, was that Agile approaches seem to remove the CIO—and the rest of IT leadership—from the value-delivery process.

For example, in Scrum, an Agile software development framework, the delivery team works directly with a product owner, who is generally drawn from the business. The product owner decides what will be valuable and works with the rest of the business to see that the value is harvested once the product is delivered. The delivery team—the autonomous delivery team—figures out the best way to deliver the solution. The team listens only to the product owner on questions of value.

Considering the above, what role does the CIO play in delivering value?

This question puzzled me, because I had previously thought that the CIO (me) had some responsibility for delivering IT value. This seemed to lead to a broader set of questions. Is the Agile team responsible for delivering business value, or is the product owner (or someone else drawn from the business) responsible for delivering business value, and the team responsible only for delivering product that will be used for delivering business value? Why do we need IT management? And if we do, how should they involve themselves in the delivery process?

What is the relationship between IT and the business, and how does it change as we introduce Agile and Lean approaches?

The more I thought about these questions, the clearer it became that to answer them, I would first need to understand some more fundamental points: How does an IT department fit into its broader enterprise? What is the relationship between IT and the business. And, how does the relationship change as we introduce Agile and Lean approaches?

There are three main places to look for answers to these questions. The first is in the literature of the CIO—the many books, conferences, blogs, and podcasts on how to be an IT leader. What I found there was pretty much nothing at all; no discussion on what it meant to be an IT leader in the age of Agility, just a great deal of speculation on how to gain a "seat at the table," or a recognition of IT's strategic value.

The second place to look is in the literature of Agile, Lean, and DevOps practices. There, I read that IT leaders had a role to play in driving the adoption of Agile practices in their organizations . . . but then what? What was the role of the IT leader once Agile practices had been adopted?

The third place to look is in our own experiences, my personal experiences and those of my friends and associates. The best I could do was to reflect on my own mistakes as I bumbled around trying to figure out what I was supposed to be doing as an Agile CIO.

I think I can promise that every point I make in this book is an answer to some moment of dumbness in my career. I love the idea that I learned about in Chris Avery's article, that in making an organization Agile one should "provoke and observe."[1] I have provoked, and I have observed.

What I have observed, mostly, is confusion. Martha Heller, in her book *The CIO Paradox: Battling the Contradictions of IT Leadership*, sums it up beautifully: the CIO role, and by extension the role of all IT leadership, is filled with contradictions and impossibilities.[2] And when I reflected on the fact that Agilists don't talk much about CIOs and CIOs don't talk much about Agilists, the reason suddenly became clear to me. The reason—I think—is that the way the CIO role is defined, conceived, and executed today is incompatible with Agile thinking.

> The way the CIO role is defined, conceived, and executed today is incompatible with Agile thinking.

There is a fundamental disconnect between the two. Interestingly, I find that the way the CIO role is defined, conceived, and executed today is incompatible with pretty much everything, particularly the delivery of business value. That, perhaps, is the point of Heller's book.

That is why this book is filled with hope. I believe that if we reconceive the role of IT leadership based on Agile principles, we can make sense of all this confusion and turn IT into a value-creation engine. Agile, Lean, and DevOps approaches are radical game changers.

They are a different way to think about how IT fits into the enterprise, a different way to think about how IT leaders lead, and a different way to think about harnessing technology to accomplish the objectives of the enterprise.

The worst thing we can do is to try to squeeze Agile ideas about project execution into a business context that was created with Waterfall approaches in mind. Unfortunately, that is what we do. The Waterfall, I will show, is so deeply ingrained in the way we think about IT leadership that we barely notice it. On the other hand, by importing Agile ideas into the leadership context, we can align delivery with management, oversight, governance, risk management, and all the other things that IT leaders have always worried about.

It seems to me that honest and open conversations are not taking place at the interface between management and Agile delivery teams. The important questions are obscured by rhetoric that says, “We need immediate cultural change so that we can become Agile!” That attitude, as I pointed out in *The Art of Business Value*, is strangely non-Agile—what we really need to do is experiment and learn about how an Agile approach to IT works within the broader business context that is the enterprise.*

That context—given the history of business management over the last few decades—is of an IT organization that is separate from *the business* and stands in a fraught and tenuous relationship with it. Agile approaches hold out the promise of solving this problem-

* I should note that Kanban, as described by David Anderson in *Kanban: Successful Evolutionary Change for Your Technology Business*, does suggest incrementally moving toward the Lean approach.

atic relationship but have focused on the micro-context of individual teams and not yet effectively taken on the macro-context of enterprise dynamics. The problem of what Agile looks like at enterprise scale is treated as a problem of how to scale Agile.

This, I say, is because we are not having honest and direct conversations. Are Agile teams saying to the senior people in their companies, "Stay out of our way! Your job is to be servant-leaders and help us!"? Or are they saying, "We are going to do this initiative without any requirements—instead, we are going to have discussions about what will create value"? Instead, I hear mumbling about user stories—yes, of course you can give us requirements; just write them in this new format. Agile teams, fearing that management will resist their Agile practices, are trying hard to frame those practices in ways they think will be palatable to management, but they are hiding critical, disruptive ideas in the process.

Are managers saying to their teams, "Yes, you are empowered and autonomous, but sometimes you produce sucky code, and not all of you are competent, and it's part of my responsibility to fix this"? Or are they saying, "Right, we shouldn't slavishly follow a plan, but no one in the senior executive team can figure out where we are on anything, and they're getting antsy"? Or, "Great job keeping the users happy—but the company's strategic goals aren't being met"? Or even just, "I feel excluded from your process"?

Honest and open conversations are not taking place at the interface between management and Agile delivery teams.

There are ways to deal with all of these issues, if they are raised openly. What I want to do in this book is to take these matters head-on, and show that they lead us to interesting places. Think of me as a tour guide pointing out highlights as we travel through the rough terrain of IT leadership today.

This book is a meditation, or series of meditations, on what IT leadership means in an Agile world. I plan to take each of the areas that we have thought of as IT leadership concerns and twist them around, look at them from odd angles, and arrive at an idea of how they appear from an Agile perspective. I will explain what I see as the fundamental incompatibility of the CIO role—as it has been defined—with Agile and Lean thinking, which represent the best ways we know for IT to deliver value. I will provide a primer on Agile and Lean thinking from an IT executive's perspective.

Ultimately, I will show that the only way to become an Agile IT leader is to be courageous—to throw off many of the attitudes and assumptions that have left the CIO meekly begging for a seat at the table, to proceed in bold strokes, and to lead the enterprise in seizing opportunities to create business value through technology.

The idea that IT is an independent unit that must be brought under control is a great destroyer of business value.

Part I explains what I see as the fundamental problem of IT leadership in the Agile world: the old idea that the primary role of IT leaders in an enterprise is to demonstrate control over an independent-contractor-like gang of technical folks, thereby justifying a "seat at the table" for IT. The idea that IT is an independent unit that must be brought under control is not only incompatible with Agile thinking, but a great destroyer of business value.

Part II is organized like many typical books for CIOs, with a chapter touching on each of the classic concerns of IT Leadership—governance, oversight, Enterprise Architecture, building versus buying, security, and so on. But there is a twist. As I take up each of these concerns, I try to show how we should be thinking about them in an Agile context if we are to have a frank and open conversation. Let's not pretend that user stories are simply a way of expressing requirements, but instead admit that there is something fundamentally wrong with the idea of a requirement as it is traditionally understood. By the end of Part II, I hope you will see why I believe that leading IT is fundamentally different in an Agile world, and thoroughly inconsistent with the traditional paradigm of gaining a seat at the table by demonstrating control over the geeks.

In Part III, I pull the pieces together to show what IT leadership does look like in an Agile world, and how IT leaders need to change their behavior in order to make the transition from Waterfall IT to Agile IT. The good news is that, by leading in an Agile way, we can tear down the wall between IT and the business and claim that seat at the table that has so often remained elusive for CIOs working within the traditional paradigm.

This book builds upon some of the ideas I presented in *The Art of Business Value*, though it is not necessary for readers to have read that book to follow the arguments in this one. In *The Art of Business Value*, I made the case that business value cannot be a guide for IT unless it is first framed—translated into a concrete set of values—by the organization's leadership team. *The Art of Business Value* examined the enterprise as a Complex Adaptive System (CAS), with emergent needs and constantly changing interpretations of business value. In this book, we are looking at how IT leaders can deliver business value to their companies with this in mind.

You will find a few themes running through this book. First, that we have locked ourselves into a frame of reference that is getting in our way as we try to become Agile. This frame of reference includes the notions of project, system, application, investment, architecture, skill set, and accountability. We have, to be honest, made a jumble of these concepts. If we sort them out and think more clearly about them, then we can resolve some of our questions about how to reconcile IT leadership and Agile approaches.

Second, that the business value of IT is more like the value of an intangible asset, which I will call—despite some disconcerting connotations of the term—the Enterprise Architecture. The asset view of IT will substitute for the outdated project view in my vision for what IT leadership must become. IT delivery is about making incremental adjustments to that asset. The asset supports the business's operations and competitive strategy; it has latent value to the extent that it will support future needs with minimal additional investment.

Third, underlying all of these changes—all of the problems with plan-driven approaches, all of the advantages of Agile approaches—is a confusion about how to deal with uncertainty and risk. What I call the "contractor-control paradigm"—our old way of doing IT—is really about trying to make risk go away, when risk is really the essence of what we do. The job of the CIO is to bring a reasonable attitude toward uncertainty into the company's decision-making process.

> The job of the CIO is to bring a reasonable attitude toward uncertainty into the company's decision-making process.

Fourth and last, that the business should be thought of as a community, or perhaps as a Complex Adaptive System, which needs to be led and managed through an inspect-and-adapt, feedback-and-vision-oriented approach because of its complexity. As a result, "control" doesn't look like what it used to, and a CIO trying to gain a seat at the table through demonstrating the old kind of control is going to set his or her company back three or four decades in its ability to compete. Reductionist management theory has led us to believe that business strategy and tactics require making plans and then executing them. This in turn has led us to make important decisions based on point-in-time snapshots of what we call data but what are really assumptions about the future. But we have learned that the pliability of software lets us test those assumptions, which leads to better decision making.

Along the way, I will focus on two critical questions:

- How can we harness Agility to achieve the best value for the enterprise?
- How can IT redefine its relationship with the enterprise to maximize this value, and in the process, earn that seat at the table?

Who This Book is For

In this book, I will be talking mainly about Enterprise IT—that is, the kind of IT that provides capabilities to users within companies, rather than digital products that are sold to customers. Digital product companies—unsurprisingly—are generally organized around products delivered to customers, a model where analogies to other types of product organizations arguably work well. When this model is applied to Enterprise IT, however, it leads to a problematic relationship in which IT as the provider of products and services is held at arm's length from its customers, who happen to be its fellow employees.

In much of the book I draw my examples from software development. I find that many of the book's themes are most vivid in the software world, and that happens to be the world where much of my experience lies. But I am really talking about all types of IT delivery capability. In chapter 8, I will specifically address capabilities acquired "off the shelf" and broaden the discussion, albeit briefly, to include hardware.

If you are a CIO, you need to not only understand this Agile revolution, but also to do something about it. This book will explain it all, and what it all means for you. I will show you why this is a deep and important change and why you need to adapt to it—and I will show you how.

If you are in another IT leadership position, your concerns are similar to those of the CIO. You play a role that links IT delivery to IT and business strategy, and you need to acknowledge the importance of the change that Agile approaches bring to both. The IT practitioners you lead have new expectations of you: you will need to manage in a Complex Adaptive System, function as a servant-leader, and remove impediments. There is a new language, and along with it, a new way of thinking. As you prepare for a CIO or more senior IT leadership role, you need to learn how Agility changes the way you create value for the enterprise.

If you are an Agile practitioner, then you have not yet figured out the role management and leadership play in your practice. I say that with confidence, as I have been part of the Agile community for more than a decade. I have not found one book that effectively connects our team-based Agile, Lean, and DevOps practices to the role that senior IT leadership plays. At best, we have told managers and leaders to keep their hands off. This book brings together the literature of Agile thinking and the literature of IT leadership.

If you are a consultant supporting an Agile transformation, I will try to give you insight into how CIOs think and a language for discussing this transformation with them. Without the ideas in this book, I am afraid you will find yourself reinforcing many of the old ideas about IT leadership—obstacles to your Agile transformation.

If you are a fan of Italian pastas, you might want to read this book because of its unique—perhaps unprecedented—use of pasta metaphors in an IT context.

Perhaps you are a senior executive in a non-IT discipline: maybe a CFO, CMO, CEO, or COO. This book will help you work with your IT organization to harness nimbleness, flexibility, speed, leanness, and responsiveness to create competitive advantages through IT.

As with all things Agile, this book is intended as a contribution, an increment, and a trial that is subject to feedback and improvement. I can't say it better than Xenophanes did 2,500 years ago: "Let these things be believed as resembling the truth."

Part I

Finding the Table

1 SITTING ALONE

I don’t know why we are here, but I’m pretty sure that it is not in order to enjoy ourselves.

—**Ludwig Wittgenstein**, *conversational remark*

Well chaps, first I’d like to say a few vile things more or less at random, not only because it is expected of me but also because I enjoy it.

—**Donald Barthelme**, *Snow White*

I’ve read a number of books on IT leadership and how to be a good CIO. None of them mention the major change of the last two decades: the rise of Agile and Lean practices for IT delivery. I’ve read plenty of books on Agile and Lean practices for IT delivery. None of them explain the role of IT leadership in an Agile world. The two domains are evolving separately: the field of IT leadership continues to frame its problems in its same old ways, oblivious to the deep changes brought on by the Agile revolution, while the Agile world, ever suspicious of management, proceeds as if it can manage without the involvement of IT leaders.

Surprisingly, this divergence continues despite the deep influence of Agile and Lean thinking on general—that is, non-IT—management. The disciplines continue to evolve separately even though corporate strategy is increasingly about both agility and IT strategy. The two worlds do not converge, even though IT leadership books advise CIOs to pull themselves closer to strategy formulation and claim a "seat at the table." But while the other C-level executives around the table are discussing the need for agility, senior IT leaders, eager to gain or retain a seat at the strategy table, are pursuing the path of demonstrating the value of IT . . . by locking in old-school practices that encourage rigidity.

Agile and Lean thinking represent, simply, the best way we know of practicing IT. The techniques of Continuous Delivery (CD) and DevOps might have originated with the so-called "unicorns"—the leading technology companies—but they have spread quickly through the "horses" to the "donkeys," dramatically increasing their deployment velocity and market responsiveness and in the process becoming table stakes for playing in competitive industries. The Puppet Labs and DORA *2016 State of DevOps Report* found that those high-performing horses and donkeys spent 22% less time on unplanned work (a proxy for quality) and 50% less time remediating security flaws, experienced 2,555 times shorter lead times, and had employees who were 2.2 times more likely to recommend their companies as a place to work.[1] The stock market bets happily on those horses, as they show a 50% higher growth in their market capitalization over three years.[2]

Admittedly, IT is *always* changing, and rapidly. Suddenly, we were delivering for desktops rather than mainframes; for client-server architectures rather than monolithic ones; for distributed abacuses, n-tier whatchamacallits, clouds, extra-large-size data, re-oriented objects, etc. Our services became microservices, apparently skipping right over milliservices on their way to becoming nanoservices. Our Businesses had Intelligences and our Internet filled with Things. We outsourced, we insourced. In this context, it is tempting to see the Agile/Lean movement as simply a buzz term that describes how we deliver IT product today.

In fact, these changes of the last 15 years are revolutionary: they are not about the mechanics of IT system delivery, but about what IT is, how it should be managed and led, and how it fits into the enterprise. Yet somehow, the literature on IT leadership and the techniques taught to current and future CIOs through books, seminars, conferences, and membership organizations continue to emphasize a decades-old, control-oriented paradigm that is inconsistent with the new Agile ways of thinking. This inconsistency, as I will show, runs deep—there are very good reasons why the CIO community is not taking advantage of the powerful changes brought on by the Agile revolution (revolution, yes—it even has a manifesto!).

Because of this divergence, senior IT leadership is pulled from one new marketing buzzword to the next, drawn to the trend of the day, while missing the deeper currents that could change the way technology is used to drive business value. Locked into an understanding of its role that involves protecting or striving for a seat at the table, practicing governance, finding cost efficiencies, executing projects against defined milestones, and delivering service with a smile, IT leadership is blindsided by IT-like initiatives it plays no part in—initiatives executed by shadow IT organizations, rogue developers, and the newly knighted Chief Digital Officers and Chief Data Officers.

The prevailing wisdom about what makes for good CIO leadership would make an Agile thinker squirm.

Indeed, the prevailing wisdom about what makes for good CIO leadership would make an Agile thinker squirm.

> As the project reaches each gate in a series, the project is reviewed with sponsors, the project team, and the project management office for progress against goals and key risks.

> Each gate calls for a go/no-go decision for the next stage of activity and funding.[3]

So say Richard Hunter and George Westerman in *The Real Business of IT: How CIOs Create and Communicate Value*, perhaps missing the point that this is a faithful description of the old school Stage-Gate or Waterfall model that Agile approaches reject. One CIO, answering the question of how to maintain control over IT in Martha Heller's book *The CIO Paradox*, says, "You do that through very rigorous architectural thinking, planning, and review."[4] The Agile Manifesto, on the contrary, says that "the best architectures emerge from self-organizing teams"[5]; its focus is on experimentation and evolution rather than on trying to plan architecture "very rigorously" in advance.

While Agile organizations increase delivery velocity on the theory that rapid feedback cycles and early delivery of value are critical, Heller advises CIOs to "understand that one of the most evolved of all executive traits is the ability to be patient, the ability to balance the need for speed with the patience to set things up correctly."[6] Hunter and Westerman seem to agree: "Successful CIOs don't skip steps, and they don't run them out of sequence."[7]

> Risk is managed not through cautious planning but through bold experiments.

But Agile and Lean approaches recommend that teams put product in the hands of users quickly and then continuously refine both the product and the team's practices, rather than waiting for perfection before starting or "moving on to the next step." Risk is managed not through cautious planning but through bold experiments combined with frequent inspection, feedback, and adaptation.

When Hunter and Westerman say that IT must demonstrate value through "on-time project delivery, on-budget project delivery, and 'first time right' application delivery,"[8] are they aware that the Agile community speaks instead of maximizing business value delivered, creating minimal viable products which are later incrementally enhanced, and even of testing in production?

IT leadership experts have struggled to express the practical implications of the changes brought on by our increasingly Agile, digital-service-driven world. George Westerman, in his book *Leading Digital: Turning Technology into Business Transformation*, encourages us all to become digital masters. Digital masters, he says, "use technology better than their competitors do and gain huge benefits . . . [they] see technology as a way to change the way they do business."[9] Well, of course they do.

Peter Weill and Jeanne W. Ross think it's important that businesses become IT savvy. "IT-savvy firms distinguish themselves from others by building and using a platform of digitized processes . . . to disengage people from processes that are better performed by machines," they explain.[10] To me, that sounds more like the slogan of the Industrial Revolution, not advice for IT leaders adjusting to the digital age. I don't disagree with these thinkers, but how exactly (or even approximately) should IT leaders make their companies IT-savvy digital masters?

I don't mean to pick on these authors—and especially not Heller—who've written much that is helpful and to the point. But the implications of the last few decades—the changes brought on by the Agile, Lean, and DevOps movements and the increasing importance of digital services—are much more profound than these easy pronouncements would indicate.

Surprisingly—and ominously—Agile thinking has gone right around IT leadership to influence non-IT executives, with books like Eric Ries's *The Lean Startup*, which makes validated learning a critical goal for the enterprise and argues for moving quickly to implement minimal viable products and hasten corporate learning. In fact, Agile and Lean approaches—which, in truth, are management techniques rather than technical practices—have spawned literature that bears on general corporate leadership. Non-IT executives can learn how to apply intrinsic motivation techniques from Daniel Pink's *Drive*, and can learn to see the business as a Complex Adaptive System—an evolving organism that continuously adapts to environmental factors and incentives set by leadership—from *The Biology of Business: Decoding the Natural Laws of Enterprise*, edited by Henry Clippinger III.[11]

The literature on autonomous teams in the workplace is substantial—Harvard Business School Press, for example, publishes Richard Hackman's classic book on the subject, *Leading Teams: Setting the Stage for Great Performances*. General Stanley McChrystal's book *Team of Teams* draws lessons for businesses from the military's increasingly agile ways of organizing to fight global terrorism. And the Beyond Budgeting movement teaches executives that the artificial annual budgeting cycle is not agile enough for corporate planning. All of these ideas have been deeply influenced by Agile IT thinking. While the writers on IT leadership are talking about the "need to be digital," non-IT leadership is already absorbing the lessons of actually becoming digital.

To further complicate matters, senior executives, and indeed everyone in the enterprise, have become more sophisticated in their use and understanding of the technology. They have high-speed wireless networks at home, smart watches and fitness bands, media streaming out of their devices and into their sensory organs. They shop online and ask Google, Siri, or Alexa when they have a question. Their standards for usability and functionality are high and climbing. Many of IT's partners and users have learned to talk intelligently about the cloud; they know about big data and predictive analytics; their wearable devices have more computing power than IT's servers had a few years ago. I mean, ordinary folks in the company have already learned just to hit the restart button on devices that aren't working right—what more can we teach them?

Any C-level executive can see that Facebook is changing the features on its site every day, while IT projects in his or her own company are still spitting out dribs and drabs on quarterly or annual release cycles. Yes, there are very good reasons why IT is run the way it is, and yes, IT leaders increasingly understand why Agile and Lean techniques are important for *product delivery* and *project execution*. But that is just the point—they are framing the new ideas in Agile and Lean thinking in terms of an old paradigm and missing their deeper implications. As I will argue later, IT leaders should not even be talking about product delivery and project execution. The world has moved on, and we should be glad of that—the old model wasn't working all that well for IT leadership.

> Many IT leaders are framing the new ideas in Agile and Lean thinking in terms of an old paradigm and missing their deeper implications.

Perhaps the most far-reaching change to consider is in whom executive leaders look to as their corporate models—whose strategies, cultures, and competitive tactics they study in business school and try to emulate. Netflix, Google, Amazon; the "unicorn" leaders of the technology world, of course. These companies are not just business role models but familiar and important to the company's executives in their daily lives. Leaders want to run their companies more like these successful technology companies, and who is in the way? Generally, it is the IT department, which is still producing more Gantt charts than useful product.

Non-IT executives are now speaking the language and technique of IT. But many IT executives are not.

If not adapting to Agility, then what *are* CIOs concerned with?

The typical book or blog on IT leadership asserts—as it has for decades now—that the CIO needs to claim a seat at the table*—that is, a place among the strategic-thinking C-level executives who report to the CEO. In her book *The New IT: How Technology Leaders are Enabling Business Strategy in the Digital Age*, Jill Dyche devotes an entire chapter to "Getting and Keeping a Seat at the Table." In the *EY* study "The DNA of the CIO," the authors say that "securing a seat at the top management table is—and should be—a key priority for CIOs," but point out that less than one in five CIOs occupy such a seat.[11] A 2016 article in *InfoWorld* called "CIO's May Finally Get a Seat at the Grown-Ups' Table" starts out "for as long as I can remember, CIOs have obsessed about getting a seat at the executive table."[12]

Some of these obsessive CIOs already have one, of course, but those who don't, according to these sources, must learn to put technology aside and develop the skills of C-level executives: financial savvy, polished communication, strategic visioning, and customer intimacy. It is interesting that this is posed as some kind of a difficulty. These writers seem to assume that CIOs are naturally all

* Note that the phrase is used ambiguously, either to mean a seat at the board of directors table or at the CEO's table of executives, who weigh in on strategic decisions.

introverted, anti-social techno-nerds. Is the invitation to join the strategic table being withheld because the CEO doesn't believe that IT is strategic, or because he or she does not believe that someone with technological savvy is capable of playing a strategic role?

How, according to these books, should the CIO go about gaining the coveted seat at the table? There apparently are a number of preconditions, according to the literature. "Show value for money before you try to prove that IT is an investment in future business performance," Hunter and Westerman say.[13] Let's think carefully about that statement. IT leadership should *first* focus on things other than future business performance (to demonstrate trustworthiness) *before* doing the things that the business is trusting IT to do (influence future business performance). Does that sound right?

The CIO must, according to the literature, "sell" the accomplishments of IT. He or she must show the ability to think and act like an executive and demonstrate that he or she can be trusted to keep the business's interests in mind—most importantly, by *controlling* the costs and schedules of IT projects. Hunter and Westerman give examples of ways CIOs can prove IT's value through measurements, including uptime, application performance, on-time project completion, and "first time right application delivery."[14] The not-so-hidden assumption here is that IT is not businesslike; the CIO must prove something, show business value, demonstrate business savvy. The CIO must *earn* a seat at the table.

As a consequence, IT leadership has been obsessed with demonstrating value by establishing control over IT project execution. According to a 2015 survey by the CIO Executive Council, 53% of senior IT leaders believe that "proving the business value of IT's contributions" is "highly important" and a further 39% believe it is "important."[15] IT leaders set up project management offices

(PMOs)—not only to *ensure* on-time and on-budget delivery but also to *prove* that such delivery is occurring. Because proving on-time delivery is the price of a seat at the table, the CIO must fight against anything that would make it harder to demonstrate that control. PMOs, for example, are encouraged to be enemies of the dreaded "scope creep"—that is, changes that the business stakeholders request when they realize that a system won't actually meet their needs as specified . . . but which might make it difficult for IT to show that it is delivering on schedule.

There is a danger that the CIO's struggle to prove that he or she is delivering value will actually *destroy* business value for the company. Because not all IT-related spending is directly under his or her control, the CIO is often forced to exert influence through policies, standards, bureaucracy, and no-saying. IT "adds value" by *constraining* solution formulation and delivery through its Enterprise Architecture standards, by *slowing down* delivery to users through its governance processes and maturity models, and by *adding overhead* through risk-averse security policies. By saying "no" to any work that would make it difficult to show that IT is under control—scope changes, exceptions to standards, newly unveiled technologies—IT is swallowing up forkfuls of potential business value.

Any IT leader who focuses on *demonstrating* value is simply wasting company resources.

I'll go further: any IT leader who focuses on *demonstrating* value is simply wasting company resources; IT leaders should direct all their focus to *delivering* value. Which of the other executives at the table puts that kind of effort into demonstrating that they are adding value? Is the CFO preparing slide shows on how drafting the annual financial statements is valuable?

The prevailing wisdom further requires that IT leadership—to justify a seat at the table, of course—demonstrate that IT is "aligned" with the business. "The root cause of most of the challenges confronting IT organizations today is the CIO's inability to lead and manage alignment, starting with IT/business alignment," according to George Lin in *CIO Wisdom*.[16] I am particularly struck by the framing in *IT Governance*: governance, the authors say, "ensures compliance with the enterprise's overall vision and values."[17] It is revealing that they think governance is about compliance: the vision and values come from somewhere else, and IT must comply with them.

> IT leaders should direct all their focus to delivering value.

Although aligning with strategies formulated outside of IT would seem to absolve the CIO of some responsibility for results, it actually doesn't work that way: the more senior a corporate executive is, the more likely he is to blame IT when things go wrong.[18] Or, as Martha Heller puts it, "there are only two types of projects: business successes and IT failures."[19]

Nor is the whole idea of earning a seat by demonstrating value very effective. The same CIO Executive Council study asked what the prevailing perception of IT was by business stakeholders. Fifty-eight percent said that IT was perceived as a service provider or just a cost center; 28% as a separate but partnering group; 11% as a peer; and just a startling 3% as a business game changer.[20] I haven't seen a study on this, but what percentage of respondents would say that technology itself—as opposed to the IT department—is a business game changer? High, I'd think. What then does it tell us that only 3% think that the IT department is a business game changer?

Perhaps the reason business stakeholders perceive IT as a service provider is that—um, well—we have defined the role of IT to be a service provider. "The CIO is responsible for the smooth running of 24-7 IT operations, IT governance, and implementation of new projects," according to Weill and Ross.[21] Funny that the role doesn't involve outcomes, isn't it? If we continue to define IT as a *function*, then we will never find a seat at the table.

> The job of the CIO is to bring a reasonable attitude toward uncertainty into the company's decision-making process.

That role is too passive—"implementation" not "formulation"; "governance" not "roadmap." Investments and projects somehow come to the CIO from somewhere else, and IT delivers on them. While pointing out that more than 70%[22] of IT spending typically goes to keeping the lights on, which is presumably non-strategic activity, the experts still advise that CIOs demand a seat at the strategic table. But if most of IT's effort is simply wiping up messes or filling the salt shakers, taking orders and delivering the dishes, and perhaps helping those at the table choose their wines, how can a seat at that table be appropriate?[†]

This passivity bleeds over into an attitude that IT's function is to serve the rest of the enterprise. Somehow, IT cannot shake the notion that its role is about providing "customer service" to the rest of the business. But why? Does finance provide customer service? Marketing? Mustn't we admit that IT leadership's obsequious attempt to charm its way to the strategy table by pleasing other executives further undermines its chance of gaining that seat? A business function

[†] No disrespect to janitorial or service-profession functions intended; my point is just that it is hard to argue that they are strategic to the company.

that merely serves the real strategy creators can never deserve a seat at the strategy creators' table.

So, the CIO is fighting for a seat at the strategy table by demonstrating basic competence, believing that the rest of the organization considers IT to be non-businesslike, potentially misaligned, and in danger of destroying business value. The CIO demonstrates basic competence by showing that IT is none of those things—that, in fact, it is delivering good service. Is it any wonder that the business perceives IT as a service provider and not as a game changer?

We can all agree that adding business value is a good thing, but a CIO trying to *prove* that he or she is doing so is in for a tough struggle. I've read—and written—CIO resumes. Sure, we say that we saved the company umpteen million dollars by automating the schmoo process or consolidating our whatsits. But, as we all know, that's assuming a particular baseline (really?) and assuming that the results were actually realized (were the former executors of the schmoo process let go, at no cost to the company?); it is ignoring the question of whether that cost-saving was more important to the company than other things IT could have been working on; it assumes that customer service and employee satisfaction didn't decline; and it especially avoids the question of whether a different CIO could have achieved even more benefit at less cost, or simply eliminated the schmoo process entirely rather than automating it. Perhaps a better CIO would have seen that the industry is changing in ways that will soon make the process irrelevant? The business value delivered is more the result of a well-written presentation than a business reality.

I agree with Douglas Hubbard's point in *How to Measure Anything*, that anything can be measured if it can be *defined*;[23] the problem here is that the definition of IT's value is simply wrong, or at best, confused. In *Leading Digital*, Westerman, Bonnet, and McAfee frame it as, I think, most people do: "It starts with competence in delivering services reliably, economically, and at very high quality."[24] But is this what we mean by IT adding value to the enterprise? Weren't we talking about a strategic function, about a digital world in which the company uses IT to compete?

There is a deeper problem with this idea of demonstrating competence, and that is the intrusion of uncertainty into the realm of IT decision-making.

I have discussed the challenge of defining business value in *The Art of Business Value*; even more challenging in this case is defining what we mean by business value *delivered by IT*. Business value is delivered by the enterprise with support from IT—IT is part of a whole, a complex system in which its ability to deliver value depends on factors outside of IT. The only way that IT can *deliver* business value itself is through cost-cutting within the IT cost structure—in all other cases that I can think of, IT is delivering *product* that might or might not then be used by *someone else* to deliver business value.

Aside from the problems of measurement, there is a deeper problem with this idea of demonstrating competence, and that is the intrusion of uncertainty into the realm of IT decision-making. The CIO is asked to demonstrate that he or she is in control of IT investments by showing that he or she can deliver on business cases according to plan. Projects should be on time, within budget, and at a high level of "quality," whatever that word might mean.

Here's the problem: plans are about the future, and the future is uncertain. In the case of IT projects, the uncertainty is extremely high. In truth, an excellent CIO is one who makes good decisions about risk and adapts plans over time based on unexpected events and changes in the company's needs. But if the CIO is trying to justify a seat at the table based on his or her control of an uncertain future, he or she will be off eating in the corner of the room or banished to a side table in the kitchen.

Let's listen to Westerman, Bonnet, and McAfee:

> In the long-distant past, we were taught that IT was the keeper of technology and that IT leaders were service-providers to the rest of the business. Their job was to stay aligned with business strategy, taking orders from the business and delivering new systems. If they kept the systems running and delivered new projects on time, then all was good. That time is over, and has been for many years.[25]

It has been over for many years. But we have not shaken the idea.

In the twenty-first century, there are very few C-level executives out there who seriously doubt that IT adds value to the business. The rest of the enterprise does not want IT to treat them like customers, and does not want IT to "align" with them. What they want is for IT to deliver *outcomes*.[26] This screams out for an Agile and Lean solution: deliver value—outcomes—quickly and frequently, and trim away everything else, since everything else is simply waste.

Getting a seat at the table is not so much about learning to wear a suit and tie instead of a Nirvana T-shirt; it's about guiding the enterprise in its use of technology and information assets. Given the importance of digital technology, and given which companies now serve as role models for executive leadership, it might be that instead of CIOs learning to wear suits, the

Instead of CIOs learning to wear suits, the rest of the executive team should be ready to start dressing down.

rest of the executive team should be ready to start dressing down. Or perhaps the business should be learning how to align with IT, rather than the other way around.

Stop laughing. This is just another way of saying that businesses need to become digital masters who are IT savvy.

So, there we have it: the CIO is being told—and told loudly—that he or she must earn a seat at the table by proving that he or she can control IT delivery—a discipline in which uncertainty is the norm—and deliver business value—difficult to define—while for the most part being restricted to delivering only product. If the CIO does these things, then he or she is probably destroying business value and doing non-strategic things, and therefore does not deserve a seat at the table. What will this CIO be doing with his or her time? Creating bureaucratic policies to try to establish "control," denying that IT is too expensive, pretending to have near-certainty in situations where uncertainty is the norm, and saying no to anything that might interfere with his or her ability to prove that he or she is adding value.

This all comes at a time when businesses want to become IT-savvy digital masters. In Westerman, Bonnet, and McAfee's research, "many executives told us that, given their IT units' poor performance, they were going to find a different way to conduct their digital transformations. The business executives were going to move forward despite their IT units, not with them."[27] Again, this bears some deeper thought. We know that IT organizations are often filled with motivated, intelligent, and experienced professionals. If particular skill sets are missing from the IT organization, they can be hired, just as they can be hired into any other part of the organization. There is no *a priori* reason that IT cannot lead the business's digital transformation. The fact that organizations widely don't believe this suggests that there is something wrong with the way we have been defining IT.

You would be disappointed if you thought to turn to the Agile community for ideas on how senior IT leadership should act. The nicest way to characterize the Agile community's treatment of senior IT leadership is, well, neglectful.

When the Agile literature does think to mention senior IT leadership, it does so in the context of driving cultural and organizational change to make room for Agile practices. It assigns two roles to IT management. The first is to empower teams to be autonomous. The second is to help drive the cultural change that will allow the adoption of Agile techniques. These both come down to the same thing: management's job is to force itself to stay out of the way.

In a way, the Agile community is suffering from the same insecurity as the CIO community. While CIOs feel that they need to justify their existence and claim a seat at the table, the Agile community is stuck on the idea that it has no place until dramatic cultural and organizational changes happen.[††] The enterprise is assumed to be naturally resistant to Agile ideas, and Agility has to fight to claim a seat at the other table—the PMO table, that is. I have argued elsewhere that the fist-pounding demand for cultural change is misguided: there is a place right now for an agile approach to Agility, cultural change or no.[28]

But the real inconsistency—or paradox, if you prefer—is that Agile approaches seem to *remove* IT leaders from the value-creation

[††] Thanks to Gojko Adzic for pointing out to me that Kanban is different in this regard—it advocates moving incrementally from the status quo. On the other hand, I hear very little of that attitude in discussions in the Agile community.

process. In an Agile process, visioning, refinement, and acceptance of system capabilities are in the hands of product experts and users—that is, the folks from the business side. Delivery teams work directly with users and product owners from the enterprise lines of business to decide what is valuable and to create solutions. Where does this leave IT leaders who have always believed themselves responsible for making sure that IT delivers business value?

According to Ken Schwaber and Jeff Sutherland, the creators of Scrum, "Self-organizing teams choose how best to accomplish their work, rather than being directed by others outside the team."[29] A product owner, generally drawn from the business, is responsible for maximizing the value of the product that the development team produces.[30] For the product owner to succeed, they say, "the entire organization must respect his or her decisions . . . the Development Team isn't allowed to act on what anyone else says."[31] It is hard to see how the team could be more autonomous: the product owner decides *what* they should do, they decide *how* they should do it, and no one else—presumably including IT leaders—is allowed to give them direction.

Now, this is considered a fairly extreme statement of Scrum's views, and other Agile frameworks describe the autonomy of the team differently. Extreme Programming (XP) is less prescriptive and speaks instead of the development team working with "onsite customers," but the general idea is still to empower the team to deal directly with non-IT "business" people to create valuable solutions. The autonomy of the team is further extended in the DevOps model, where a team has "full stack" responsibility for a product: not just development, but also testing, operations, security, and infrastructure engineering.

Let's face it, in this world, IT leadership can sometimes become a useless barnacle on the ship of value delivery, a parsley garnish on a bowl of chocolate pudding, an elephant-shaped stapler on the dinner table.

Imagine the ancient Greeks with their autonomous city-states. They found a way to govern locally through direct democracy. Of course, the city-states were autonomous—you might even say that they worked at cross-purposes. There was no real vision of a single Greek nation that brought those city-states together. Now imagine the Romans with an altogether different vision. Initially a republican vision, theirs was ultimately a vision of empire; their governance structures were set up to unify diverse city-states under a common vision. They had an enterprise view. The Greeks, with their loose collection of autonomous teams, could not compete and eventually were amalgamated into the Roman Empire. Or would you say that the Greeks were the ones who were successful, given that their culture was so vibrant and considering all of their achievements in philosophy, drama, science, and math? Not to mention the fact that, when the Roman Empire declined, what was left was a more-or-less Greek Byzantine Empire? Well, the battle between Greeks and Romans continues.

Our poor IT leaders are Roman emperors trying to find a job in a Greek civilization. It's not surprising that 91% of senior IT leaders think that their job is becoming harder.[32]

When IT leadership finds itself separated from the day-to-day creation of value but nevertheless has responsibilities—security, Enterprise Architecture, cost control, reporting on accomplishments, switching from Python to Ruby and back again, sounding good in front of its peers at conferences—it asserts its control through bureaucracy.

Mike Cohn's wording on this subject is telling. "A Scrum team's job," he says, "is to self-organize around the challenges, and within the boundaries and constraints put in place by management."[33]

Management's role is to *constrain*, to put *boundaries* on the team's ability to create solutions. Cohn is right, though I don't think this is quite what he means. How else can management have influence on a fully autonomous team that is not supposed to listen to them other than by setting up a bureaucracy of rules and constraints? IT security formulates controls and demands that teams produce documentation to show compliance. Enterprise Architecture develops standards that the project teams must follow. The PMO adds paperwork to project reporting and to quality assurance gates. IT leaders write their value delivery story in policy papers rather than in software code.

> IT leaders write their value delivery story in policy papers rather than in software code.

What Cohn meant, I think, is that management *defines the problem* by indicating its boundaries. The team, after all, needs to know what problem it is trying to solve, and what sorts of solutions will be considered effective in solving the problem. We might better say that management sets the *criteria* for success. I admit that is a good point, Mike, but I'd rather take your words out of context for a moment.

Even granting Mike Cohn the right to mean what he means, in Scrum it is actually the product owner, the person drawn from the business side rather than IT, who defines the boundaries of the problem. IT management is relegated to the role of policy-writing bureaucrats who still control that constraining, no-saying, wonkified gate to production. Tom Demarco et al. talk evocatively about template zombies.[34] I love that formulation. IT has been biting itself in the neck for decades, and can't seem to stop. Given the several generations of IT we've been through, our templates have had baby templates by now, and this population explosion has placed severe demands on the scarce resource of delivery team time.

The Agile movement is undermining itself by—in effect—encouraging senior IT leadership to fabricate bureaucracy and manufacture constraints. It does this by not admitting that there is a continuing role for IT management even after the Agile transformation is accomplished, by forcing the CIO to eat brains. It is interesting that both the CIO literature, with its obsession with earning a seat at the table, and the Agile literature, with its emphasis on autonomous teams, both wind up compelling IT leadership to destroy business value.

Yes, I am saying that Agile development adds bureaucracy and waste. #CheekySmile.

The thing is, Agile and Lean approaches work—they lead to good outcomes for the enterprise. They provide excellent ways for IT leaders to lead. They reduce risk, improve quality, and most importantly, they are agile—they allow companies to change quickly and respond to a changing competitive environment. Lean approaches eliminate waste and shorten delivery times. Who could argue with that? Agile and Lean ideas are good things for us IT leaders. The changes we are seeing are positive changes.

My fellow IT leaders, we must use these new Agile, Lean, and DevOps practices as a lever for changing the relationship between IT and the rest of the business. We have defined our roles and our goals in ways that are inconsistent. We simply cannot earn a seat at the table by doing the things we believe we need to do to earn that seat at the table; we simply cannot interact effectively with others at the table even if we have been given a seat. We are locked into a way of co-existing with the rest of the enterprise that is based on old stereotypes and assumptions—both about IT and about the business—and that destroys business value.

We can change this! The ideas behind Agile approaches are potent and compelling, and can help us reframe these business-IT interactions in a way that will create value—lots of it! Let us question the constraints, define the problem, brainstorm solutions, turn them into hypotheses, test them, and continuously improve how we practice IT leadership, as Agile thinking teaches us to do.

I believe that this is simpler than it sounds. It is about identifying the obstacles in our way and taking today's best-practice ideas—those found in the Agile Manifesto and in books like *Lean Startup*, *Lean Software Development*, *Lean Enterprise*, *The DevOps Handbook*, and others on today's management bookshelves—and applying them to IT leadership.

The news is good, colleagues. By the time you get to the last few chapters of this book, you will see that these changes in the IT world give you fairy dust and wizardry, new powers and influence that can be wielded for the good. What you must do now is open your mind, maintain your curiosity, and take on these new challenges courageously!

> Obsessed with proving that it deserves a seat at the table, IT leadership continues to frame its problems in the same old ways—oblivious to the deep changes brought on by the Agile revolution—while the Agile world, ever suspicious of management, proceeds as if it can manage without the involvement of IT leaders.

2 KEPT FROM THE TABLE

> A picture held us captive. And we could not get outside it, for it lay in our language and language seemed to repeat it to us inexorably.
>
> **—Ludwig Wittgenstein**, *Philosophical Investigations*

> The almost insoluble task is to let neither the power of others, nor our own powerlessness, stupefy us.
>
> **—Theodor Adorno**, *Minima Moralia*

In the beginning, there was technology. The company's technologists knew things that the rest of the enterprise didn't. They needed a leader, one who could communicate with the laypeople and yet represent the geeks. The CIO role was created—some 35 years ago as I write these words.[1] But the role was without form and void. It needed maturing and definition. It was not enough to say that the CIO was the geek who wears the suit; after all, that didn't help the other executives understand how to work with him or her, and it didn't really tell the CIO how to behave and what to spend time on, aside from hanging with the geeks and doodling FORTRAN code.

The truth was that the IT organization was filled with all manner of strange creatures walking upon the earth. Engineers, but not engineers who created product for the company to sell, your whiz kid inventor types. No, they were engineers who had something to do with actually making the company run. You had to rely on them so you could do your job. But, like most engineers, they were smart, quirky, and spoke a funny language. In the early days, you might even see them in breath-fogging, air-conditioned "computer rooms" wearing white lab coats and carrying reel-to-reel tapes.

They made you feel stupid. They were unpredictable. They were somehow *other*—while you were focused on business outcomes, they were focused on . . . what? They got all excited by making the computer do strange, seemingly useless little things. They took forever to get the simplest tasks done. They babbled away in acronyms and tossed cute but baffling little techie terms around, always with their peculiar sense of humor.* They were arrogant and looked down on people who didn't understand the difference between a linker and a loader. They cultivated aloofness. They were definitively *other*.

It's not enough to say that the CIO is the geek who wears the suit.

The CIO, then, was the executive responsible for *them*—the bridge between the worlds of the real business people and strange engineers. "It is commonly accepted that a CIO's value to an organization comes from the ability to bridge the gap between information technology and business."[2] Two obvious questions arise, since a bridge needs to be anchored on both sides. On the business side, what role should the CIO play? A provider of customer service to the business lines? An enforcer of policy and standards? A reassuring voice of technology familiarity? On the technology side, was the CIO a technologist leading technologists, a pointy haired management suit allocating budgets and blamed for schedule overages, an architect of the company's information function, or an accountant of IT chargebacks?

* For example, "GNU Unix," a recursive name where GNU stands for "GNU's Not Unix" Ha! Get it? Pretty funny, eh, business customer?!

The first of these questions begat a literature: advice books from experienced IT leaders and business authors telling senior IT leaders how to act like real business people to justify their role and importance in the company and to frame problems in business terms rather than technical terms. The advice to senior IT leaders sounded something like this: Get a seat at the table! Act like a C-level executive! Communicate the value of IT! Speak the language of the business, and only the language of the business! Drop all the techie stuff!

But it was on the second question that the CIO's real role became apparent. The CIO was there to keep them *under* control. Make them deliver business value, at a reasonable price, without any strange, geeky games. Maybe get them to wear freshly washed clothes now and then, and sleep at home rather than on their desks. *They* needed to be *controlled* because they were different, and thus untrustworthy. IT things—whatever you call that stuff—took much too long and cost too much: clearly, the IT people, instead of focusing on profits, were fooling around with the technology just because it amused them. They were out of control, or would be unless senior IT leaders could rein them in.

Honestly, the unpredictability and opacity of IT drove the other C-level executives crazy. It was easy for them to blame their frustrating lack of control on the technologists themselves, with their funny T-shirts and lack of business polish. Fortunately, the CIO would earn his seat at the table by showing that he could control *them* by making them deliver with predictability and teaching those arrogant folks to treat the rest of the company as their customers.

Thus, a distinctive way of thinking about IT was born, and has determined the course of IT since. First of all, we came to speak about "IT and the business" as two separate things, as if IT were an outside contractor. It had to be so: the business was us and IT was *them*. The arms-length contracting paradigm was amplified, in some companies, by the use of a chargeback model under which IT "charged" business units based on their consumption of IT services. Since it was essentially managing a contractor relationship, *the business* needed to specify its requirements perfectly and in detail so that it could hold IT to delivering on them, on schedule, completely, with high quality, and within budget. The contractor-control model led, inevitably, to the idea that IT should be delivering "customer service" to the enterprise—you'd certainly expect service with a smile if you were paying so much money to your contractors.

> We came to speak about "IT and the business" as two separate things, as if IT were an outside contractor. The business was us and IT was them.

IT and the business—two separate entities, with the poor CIO trying to keep one foot in each world while awkwardly struggling to gain a seat at the big table where the business executives sat. The business figured out business needs and handed them to the CIO, tapping its foot impatiently while it waited for him or her to deliver results. When IT (finally!) finished building or acquiring something to meet those needs, they turned it back over to the business, smiling and waiting for a pat on the head. If the CIO did well enough to get plenty of head pats, then perhaps he or she would have a place at the strategy table.

Stereotypes emerged, solidified, and remained unquestioned. If left to its own devices, IT would diverge from alignment with the business. IT people would play with the technology; do things that added no business value. IT people did not really understand what the business needed, and were incapable of making good business decisions and trade-offs. The business, for its part, was clueless, full of politics, and apt to point fingers. And, oh, yes: IT was just too damn expensive.[3]

How could you control a contractor? You asked for an estimate and pressured the contractor to deliver at or close to that estimate. Or you agreed on a fixed price. How could you control IT? Same model, but with the twist that the IT staff were your own employees who were paid a fixed salary—a bit awkward. Since their cost was fixed (at least in the short and medium terms), your biggest worry was that they would waste time on frivolous activities. How could you know that they weren't? Simple: you insisted that they deliver on schedule, and kept the pressure on them to do so.

Some of IT's work was *transactional*: user support, device provisioning, updates, and maintenance. In those areas, costs and lead times could be benchmarked and monitored. But a good deal of IT's work involved *delivering capabilities*—developing and integrating applications, rolling out ERP systems, installing collaboration tools, and so on. For IT to demonstrate that it was performing that type of work responsibly and for the business to verify that it was doing so, the scope of each task had to be defined precisely, bounded, and agreed upon in advance. The work had to be organized into *projects*, which are units of work with a defined set of deliverables, a beginning, and an end. You could establish control by making sure the project was completed within the bounds of its estimated cost and schedule. How perfect the Waterfall model is for this purpose! How perfectly it aligned with the business's need to know that IT was under control.

Of course, there are some problems with this way of organizing IT's work. In fact, when you really think about it, it makes as little sense as a semicolon terminator in a line of Python code. It could only have been born out of unease about the "black box" of IT and its seemingly uncontrollable costs.

For one thing, how does the idea of "project" fit with the idea of a "capability," an "application," or a "system"? Projects can be scoped, started, and ended; planned and measured against that plan. IT capabilities, on the contrary, are long-lived, granular, evolutionary, in constant flux, and future-oriented. After the project to build a system is finished, well, somehow the system keeps costing money! There always seems to be more work to do on systems that have already been finished. How many a frustrated CEO has complained that IT costs never end?

Aha! We found a way! We would fix the scope of the project all the more rigidly by defining some deliverable as the final operating capability (FOC), and then throw everything that happens after FOC into a bucket called operations and maintenance (O&M). Then we could try to minimize it. This in effect treats the system as a "product"—once the product is completed, it is done, and all that has to follow is the cost of "maintaining" or "operating" it. Like a car, right? But, alas, this is not how the world works. More or less secretly, the IT folks were actually enhancing and changing the product after it was finished—not just doing "maintenance" on it—to meet the business's needs as its competitive environment changed.

Unfortunately, the more rigid the scope was, the less likely the system was to meet actual business needs. A rigid scope resists change, while change in the business environment is constant. It also allows no room for errors in the original specification. While it might seem like a good way to manage contractors—that is, IT—it is also a good way to destroy business value. But at least it met the underlying goal—controlling the strange IT folk, who might suddenly burst into some peculiar, torpid yet manic activity bent on wasting the company's money.

IT was an island separate from the business; the CIO would need to control the natives, work the mines, and export the gold and silver. Or IT was the leper colony, and the business didn’t want to get too close, preferring to communicate by tossing requirements documents over the channel that separated them.

IT and *the business*. To quote Martha Heller again, “Rather, the ‘and’ in ‘IT and the business’ connotes separateness and difference, an ‘us and them’ perception that has plagued IT organizations since the beginning of their existence.”[4]

IT and the business have long known that they were in an uncomfortable, codependent, abusive relationship. To the business, Arthur C. Clarke’s law holds: “Any sufficiently advanced technology is indistinguishable from magic.”[5] IT stuff is alchemy—an esoteric art practiced by pocket-protector sophists who can run circles around anyone foolish enough to argue with them. And the IT folks—they knew they could turn on the gibberish and obfuscation any time they wanted.

IT stuff is alchemy—an esoteric art practiced by pocket-protector sophists who can run circles around anyone foolish enough to argue with them.

When the business created a project, how could it know what a reasonable price tag for its set of requirements would be? Imagine yourself at a market in rural Codeistan trying to buy a fancy snow-globe souvenir or a Sergey Brin bobblehead, and there are no marked prices. You are sure you are being ripped off. The merchants can see that you are a tourist and have no idea what the prices “should be.” You also don’t quite get that there is no price things “should be”—prices vary depending on the circumstances.

Whatever price the vendor names, you cut in half and start the negotiation from there. Whatever price you wind up with, you still feel cheated.

The IT people were asked to estimate the cost and schedule for a project in advance. Since no one outside of IT had a good basis for judging whether the estimate was reasonable (putting aside the question of whether IT itself knew), the business folks felt that they had to negotiate the estimate—the cost was always too high, of course. The best the business could do was to refer to what seemed like comparable projects, possibly unrealistic quotes they had received from external vendors, a threat to outsource if the IT folks didn't reduce their estimate, and brute-force tactics like "let's cut it in half and then add a margin in our own minds because we know IT is always late."

Like the tourists in the Codeistan marketplace of the IT geeks, the business leaders left feeling ripped off. "IT is too expensive!" they said. Of course, even though it was "too expensive," they continued to buy it—as Tom DeMarco points out in *Why Does Software Cost So Much?: And Other Puzzles of the Information Age*—because they still expected to get enough value to justify the cost.[6] The tourists might complain that they are being cheated, but if you don't bring back a bobblehead and snow globe, no one will believe you've been to Codeistan. It still doesn't feel very good.

You know how this game goes: IT would learn to pad its estimate ("management reserve," "slack time," "contingency"), and the business would knock it down to something that made them feel like they were winning the negotiation. "No, we can't possibly wait a year for that capability. You have to do it in six months."

Unfortunately, these internal-to-the-company estimates are *estimates*, not prices. You don't get a better deal by knocking down an estimate. But it seemed so, because soon, the estimate would be accepted as the plan—the schedule, the budget. Control was achieved by telling the geeks to deliver on the estimate and holding them *accountable* for doing so. A project that didn't meet its estimates "by definition" must be out of control, troubled, and staffed with lazy or incompetent people. The distinction between an estimate, a target, a plan, and a measure of success disappeared behind the Waterfall—disappeared into an illusion of control.

Now that the estimate was locked in, it was time to begin the first phase of the project—gathering requirements. The requirements would be "specified" by the business without reference to what was possible, natural, or easy; specified, and then tossed over the wall to IT. The business learned a very important lesson: if the scope of the requirements increased while the project was underway, then IT would have an excuse for exceeding the planned cost and schedule. Demanding execution against plan, remember, is what gives the business control over the IT delivery process. The enemy of control, it follows, is *scope creep*: the gradual introduction of new requirements after the plan is created. Both the business and IT could agree that this is undesirable.

What behaviors does all of this incentivize? The business is motivated to get the requirements perfect at the outset; if they make any changes, then IT will force them to acknowledge that they are setting the project behind schedule and over budget. So the business tries to make sure that everything it *might* need is part of the requirements. Into the mix are thrown wish lists of features drawn from across the user base; features to support plans that the business has in mind but that might never get funded or executed; animated cats that stroll across the screen; ideas for things that plausibly sound like they *might* add value.

The result: *feature bloat*. Everything we *might* need rather than what we *do* need. More requirements than expected in the original estimates. Why not? The estimate had already been given, so the additional scope was "free" to the business representatives informing the requirements document. The geeks themselves made things worse by pushing the business folks to specify their requirements in ever greater detail so that there would be no ambiguity. Of course, with bloated scope, it also took longer to finalize the requirements, document them, and get them approved. The schedule and cost were already compromised even before work began.

Guess what: the results were not good. Somehow, despite the fact that IT Island was populated by brilliant, hard-working (okay, albeit strange) people, virtually every project was over budget, behind schedule, and filled with defects. IT's customer service skills weren't all that great, either—they liked to say "no" and insisted on using technical mumbo-jumbo. And wouldn't you know it—the geeks always had some excuse for this bad performance, especially the schedule slippage. The requirements had changed, they'd say. The hardware broke. The operating system had a bug. Memory was leaking. The jabberwocky had a slithy tove.

The geeks were still out of control!

"Knock, knock."

"Who's there?"

"CIO"

"CIO who?"

"It depends on your requirements."

"Er, um—could you come back later? We're eating now."

"Could I join you?"

"Uh, no. We don't have any seats left at the table."

Don't think this "control" attitude was one-sided. The geeks responded with all the tricks they knew for controlling the business so that they could deliver on their accountabilities. They "required" the business to specify its requirements fully before they were willing to engage. To keep costs under control, they established standards—that is, they controlled what the business could do and could purchase. In other words, they said no. To deliver on their accountability to keep the company secure, they established security policies and controls. In other words, they said no. To maintain the integrity of the data in their databases, for which they were accountable, they "idiot-proofed" their systems, constraining what data could be entered and when. Even their software said no.

This last point is especially significant—the business folks are "idiots" to be talked down to and controlled in return. Mustn't use technical words, mustn't assume they know anything. If you want to control us, we will have to control you back.

These are interactions born of desperation, of deep-seated fear of the technologists that has been cultivated by the technologists themselves. That's why I called this a codependent, abusive relationship. Because of a cultural discomfort between the business and *them*, and the resulting dynamic of control and counter-control, a vicious cycle was engendered. But IT projects are—by their nature—not amenable to this sort of control. They are highly complex and unpredictable, influenced in a large way by small outside influences. In its desperation, the business evolved a set of practices that actually work *against* its best interests, leaving IT leadership in the middle of a mess of contradictory incentives and impractical demands.

As IT was incentivized to provide service with a smile, the business was disincentivized from learning and adapting to the new technology landscape. Consider technical jargon, for example: the IT staff was discouraged from using it because it would make business customers uncomfortable. But should IT really refrain from using technical terms any more than marketing should stop talking about branding and finance stop talking about assets and liabilities? Should we really assume that the non-IT folks are incapable of learning some of the important language of IT? Isn't jargon, jargon because it is expressive and concise once understood?

Isn't jargon, jargon because it is expressive and concise once understood?

Perhaps these questions are moot—technology has become so central to everyone's lives and so consumerized, that most of the company *does* speak the jargon these days. People store their photos in *the cloud*, distinguish between Mac, Windows, and iOS *operating systems*, and might even know what a *domain name* is. The real problem is not the jargon, but the use of it for obfuscation.

To avoid making the non-IT folk feel stupid, IT responded, over and over, to password reset requests, questions about what button to push to make the phone vibrate, CD drives being used as cupholders,† claims that the internet is broken, difficulties in watching their porn videos, and mousepads alleged to be incompatible with the company's computers.

IT does so in a cheery, bright-eyed way, as if to earn tips. But these requests are a cost to the enterprise. As the world becomes more digital, shouldn't users be expected to become more sophisticated in their use of technology and its terminology?

When it comes down to delivering capabilities, a customer-service-oriented IT organization must give the business what it *wants*. In an Orwellian twist of language, we refer to what the

† May be apocryphal, but the story has been around for a while. In any case, I've seen worse.

business *wants* as what it *needs*. That's why we have *requirements*. Isn't this a strange way to speak to colleagues in your own company—"this is what we require of you" or "this here is what we need"? Never mind that IT might have deep knowledge of the business's needs born of its long experience, deep involvement, and ability to see across all lines of the business. The job of IT has been to deliver, not to decide what needs to be delivered. The business understands business value, and it's just a matter of translating it into terms the IT folk can understand.

While the IT organization is expected to develop standards and policies to reduce costs and keep the company secure, actually applying them—saying "no" with a frown—is bad customer service. Who wants to type a long password when they log in to a system? IT forces us to. Why do we have to standardize our desktop models and configurations? IT forces us to. The business might imagine features that would turn out to be prohibitively expensive to build. Is it IT's role to satisfy these "needs"? Why does the business have to accept systems that don't quite live up to their fantasies? IT forces them to.

According to Lyndon Tennison of the Union Pacific Corporation (as quoted in Heller), "The fewer tools I have in my tool kit, the more cost effective I can be. If I can force standardization through an architecture model, I should inherently be able to drive efficiency."[7] Note the word "force." The difficulty here should be clear: IT is expected both to enforce standards and to provide customer service—that is, fulfill the customers' desires while at the same time discouraging those desires. Enforcement with a smile. Perhaps this is something like the police breaking down a door and then distributing a survey to see whether the occupants are happy with the service they've received.

The business, of course, does not speak with a single voice. Different parts of the organization may want different things (or at least prioritize different things). IT is sure to disappoint some of these divisions of the organization, given limited resources. Sure, the company might set up governance processes to resolve these conflicts, but only at the level of deciding among projects—while IT will still have to deal with conflicting opinions and desires as the projects are being executed. And why can't IT do everything, or at least more than it is doing now? The more IT tries to control costs by limiting its resources, the more unhappy its customers will be.

A subtle problem with the customer service model, which will become important in later chapters, is the impossibility of adopting an enterprise view while being bounced about by individual customer service demands. Again, Hunter and Westerman nail it: "Over time," they say, "setting up IT as an order taker produces the complicated, brittle, and expensive legacy environments."[8] We can't have it both ways: we can't expect to build a set of Enterprise IT assets that have strategic value if we are drawn this way and that by demands for service.

In any case, Agile both demands and provides a new way for IT to interact with the enterprise. And while IT can no longer be held to an impossible standard, it also can no longer hide behind that standard.

The problems with the customer service model are legion. As Hunter and Westerman put it:

> Saying that "the business is IT's customer, and the customer is always right" seems like a good idea when there is deep dissatisfaction with IT that stems from a long history of unreliable service. But over the long term, this value trap sets up the IT unit for failure because customers are often wrong (especially

> about matters in which they are not experts), and calling colleagues "customers" puts a wedge between IT and the rest of the business.[9]

To Hunter and Westerman, the customer service model is a "value trap" because it sets up a frame of reference that actually prevents IT from achieving its potential value.

> Value traps create barriers between IT and the rest of the organization, forcing conversations about IT into avenues that inherently reduce IT's value or place limits on how much IT can improve value. Value traps are often ingrained in the heads of IT leaders as well as business executives as basic underlying assumptions about the relationship between IT and the rest of the business.[10]

That is well put—this value trap is in the heads of both business and IT. Following this model, IT will get lost trying to find its seat at the table.

A customer service model presupposes an efficient market for services: the business would ideally benchmark its IT department against other offerings in the market to get some handle on what "good" performance looks like. You know, what prices they should be paying in the Codeistan craft market. Now, logically speaking, it should be difficult for the IT organization to lose a battle against outside providers: its incentives are aligned with those of the business; it has no need to earn a profit; there are no transaction or legal costs involved in transacting; the IT department already knows the business well and has no learning curve; and the IT department is fully dedicated to the customer's project. There is no *a priori* reason to think that the company's IT organization can't hire and manage people as well as a contractor. Yet somehow the IT department often comes up short in these comparisons.

Why? For one thing, we do not have an efficient market. There are no "identical" commodity services here. And besides, the comparison is also between estimates or predictions, not actuals. The internal IT organization has a good sense of what will need to be done, while an outside contractor probably does not—most likely, they are actually "bidding" on different services.

Why not simply fill the IT department with the finest people around, train them aggressively, and stop worrying about how they compare to outside firms?

Maybe we have it all backwards. Now that digital capabilities are so critical to the company's competitive strategy—to its value-creation activities—perhaps we should be designing business operations in a way that makes those digital capabilities more effective. Maybe the IT folks should formulate the company's strategy, and the rest of the business should do what is required as set forth by the IT folks to make that strategy successful. Perhaps the business should be providing customer service with a smile to the IT organization.

I jest. That was just the IT-savvy digital masters thing again. My point is that no part of the enterprise should be at arm's length providing customer service to the rest of the organization. Anyway, the business has no reason to smile: they are not having as much fun with the technology as we know the technologists secretly are, playing Warcraft and sunning themselves in LED light in those basement cubicles wherein they live.

If you are in a business environment today, look around. Try to do it with fresh eyes. Do you see IT people who are more interested in playing with technology than in supporting the business? Are the IT people still some mysterious fraternity (all men, by the stereotype) in white lab coats and super-air-conditioned rooms, speaking an obscure language that sounds suspiciously like Klingon? Is the business clueless about technology? Does the business always know what will create the most business value, and does it always follow through and harvest that value once a system is launched? Do IT projects ever really end? Do we build systems right up until we achieve FOC and then just do little bits of maintenance on them, like a car?

I might be moving in unusual circles—but it seems to me that most of the technologists I work with are human beings who are dedicated to the success of the company and eager to add business value. Not only that, but they have a deep understanding of what is valuable to the business. They know the details of the business's operations because they get to see operations and priorities across all of the business units. They know the business's challenges; they maintain relationships with all parts of the enterprise. They are more and more diverse; hailing from many cultures, they bring different sets of experiences and they have different types of communication skills. They often have political savvy, because they need it to do their jobs.

> It seems to me that most of the technologists I work with are human beings who are dedicated to the success of the company and eager to add business value.

You might also notice that the people who like to play with the technology, arguably, are now "on the business side." Subtly, without anyone noticing, it is no longer that IT needs to speak the language of the business, but rather that the business is coming to speak the language of IT.

In fact, the geeks have broken out; they've left their island and overrun the mainland. "The eruption of open source software into the mainstream in 1998," says Eric Raymond, one of the most eloquent theorists of the open source movement, "was the revenge of the hackers after 20 years of marginalization."[11] He continues "twenty years of living in a ghetto—a fairly comfortable ghetto full of interesting friends, but still one walled in by a vast and intangible barrier of mainstream prejudice inscribed 'ONLY FLAKES LIVE HERE.'"[12] By now, the mainstream world has become more comfortable with technology and has found common ground with the geeks.

Ironically, despite this convergence between IT folks and business folks, "the more your employees love technology, the more they dislike IT," as Heller puts it.[13] How could that be? "One of the reasons IT often gets such a bad name with other employees is that we always come up with complexities and barriers."[14] IT folks are no longer the *other* because they *love* technology; they are the *other*—and frustrating and annoying—because they *constrain* technology. They are the voice of the company's "control" over technology.

While the CIO and his IT legions have in many ways become the very barrier to agility, the company more and more needs to compete in a digital economy. Who should lead that effort? Clearly not the CIO. So companies have begun to create new roles for digital services experts—ultimately a Chief Digital Officer. The Chief Digital Officer's role (or that of the Digital C-suite or the Digital whatever) is to play the strategic role—the one that justifies the seat at the table—that the CIO had been vying for unsuccessfully. The CIO created a trap for him or herself: success, defined as controlling the iron triangle,[††] eliminating scope creep, taking orders from the business,

[††] The boundaries of cost, schedule, and scope. Given the uncertainty in IT plans, it is axiomatic that one can't work within all three constraints simultaneously.

and keeping the lights on effectively and cheaply, meant not being qualified to lead the company into the digital era.

It turns out to be extremely difficult to stop thinking of IT as something separate from the business. It is built into the very way we speak, the terms we use every day. In Wittgenstein's words, "The picture holds us captive." To speak of "requirements" is to imply that the business is telling IT what it *must* do; to discuss "alignment" is to imply that IT is separate enough to become misaligned. Our entire discourse on IT presupposes a separation. There is nothing analogous to this when we speak of marketing, sales, or finance. "We're becoming essentially an IT consultant to the business, climbing the value chain," say Hunter and Westerman.[15] Really, we are struggling to become *part* of the business so that we can claim that seat at the table—while continuing to hold ourselves at arm's length.

But the old models are breaking down. IT is being driven deeper and deeper into the heart of the enterprise. IT *is* the business. The business *is* IT. We do not have a "telephone" department that is responsible for the company's telephone strategy, nor a paper-and-pencil department responsible for innovation around writing. These things have become normal, operational, quotidian. As the authors of *IT Governance* put it, "In the future, describing how much an enterprise spends on IT will be meaningless. IT will be imbedded in every process and budget, just like capital."[16] "In high-performance organizations today," say Jez Humble, Barry O'Reilly, and Joanne Molesky in *Lean Enterprise*, "people who design, build, and run software-based products are an integral part of the business. They are given—and accept—responsibility for customer outcomes."[17] Or, as Heller frames it, "CIOs who have broken the paradox do not think of their role as to support and enable the business; they—and their organizations—simply are the business."[18]

IT is no longer *other*, and the objective of "controlling" IT conflicts with today's objective: to derive business value from IT by using it to drive competitive strategy, profit generation, and mission accomplishment. The wall between IT and the business can no longer stand while requirements are being tossed over it and deliveries are being made through the door. Our metaphors are mingling uncomfortably.

Agile thinking gives us a way to escape from the old I-control-you-and-you-control-me-but-we-keep-smiling dynamic with supportive, positive, solution-oriented human interaction. IT is not about obsequious salespeople who bow to a customer's demands. I would replace "service with a smile" with "interaction with a smile," or maybe "partnership and teamwork with a smile," and we should all do it—IT, marketing, operations, and even the board of directors.

It will be a good topic of conversation at the table when we claim our seats there.

The relationship between IT and the rest of the business has been defined in the same terms as that of a contractor to its customer, where the business negotiates terms with IT and then frets about its ability to control IT's delivery and customer service. This model is also called the *Waterfall*.
