VINOD SANKARANARAYANAN

SOFTWARE OWNERSHIP TRANSFER

EVOLVING KNOWLEDGE TRANSFER FOR THE AGILE WORLD

Praise for *Software Ownership Transfer*

"There are many shifts occurring today as companies implement their digital business strategies. One aspect of this shift—companies are adjusting their approach to outsourcing in part by reinsourcing systems critical to their new technology-driven strategies. The problem is that there is little research or information available on transferring application systems knowledge and ownership from one organization to another—until now. In *Software Ownership Transfer,* Vinod draws on his experiences, large and small, in making such critical transfers. I particularly like his distinction between knowledge transfer and ownership—not ownership in the legal sense, but ownership in the sense of a team taking 'ownership' of their new responsibility. Knowledge transfer is only the first step in ownership transfer. If you are contemplating the transfer of an application system—from a vendor to in-house or from one internal location to another (as happens often today)—then you need to use this book as a model for making your transfer a success."

—*Jim Highsmith, author of* Agile Project Management,
coauthor of The Agile Manifesto

"Software is becoming increasingly central to most modern organizations. Indeed, it has become trite to observe that one should no longer have a 'digital strategy' or a 'technology strategy': These strategies now form the core of the business strategy itself. This shift is seeing many organizations reconsider their software-sourcing strategy, such as, for example, bringing back in-house resources that they previously outsourced. The process of transferring the 'ownership' of core digital products is complex and can frequently be the source of disappointment, to say the least. In the worst case, significant investments can be lost in the transfer process. In this book, Vinod Sankaranarayanan draws upon his significant experience in digital product ownership and transfer, backed by copious real-life examples, and draws some important recommendations for any leader or organization overseeing a transition between teams, whether internal or external. A highly recommended read for any leader who is contemplating or is in the process of a digital product transfer."

—Chris Murphy, group managing director, Europe,
Middle East, and South Asia, ThoughtWorks

"Vinod has written a compelling book on a topic that is generating a lot of interest today but whose business outcomes are not yet clear. Combining his own rich experience as a practitioner of Agile techniques with a lucid narrative style, Vinod provides many insightful perspectives to address the challenging yet essential function of ownership transfer of IT projects. More importantly, Vinod has been able to bring structure and practical application to an area that most often than not is ad-hoc and ambiguous. This piece of work is an important step in addressing a question that will become more and more relevant in the complex world of IT outsourcing and offshoring in the years to come."

—Rizwan Hazarika, CIO and cloud advisory services lead, ASEAN region, IBM

"Knowledge transfer is an area that too many executives are willing to throw under the bus (along with QA) when budgets and deadline pressures begin to loom. Vinod provides a compelling rationale for understanding why knowledge transfer needs to be planned as a normal part of the SDLC. We ignore his analysis at our own peril!"

—John Peebles, senior vice-president, digital media at Sothebys

"Every once in a while, the care and feeding of software assets changes hands in enterprise IT. This may be because of a decision to outsource, insource, or simply switch vendors. The quality of the handover is crucial, as Vinod points out in this one-of-a-kind book. But handovers receive less attention than required—in practice as well as in theory—as evidenced by the lack of books on this topic. Handovers are typically dealt with as a three-month transition exercise, usually irrespective of the size and complexity of the transition. Post-transition performance almost always dips as a result, and it takes years to recover. *Software Ownership Transfer* provides excellent insights on what it takes to avoid post-transition blues."

—Sriram Narayan, author of Agile IT Organization Design *and IT management consultant at ThoughtWorks*

"Typically in a project-transition context, the environment is hostile as one team is losing the work and the other is taking it over, which leads to not-so-productive outcomes. Vinod has brought in a new approach to the situation, using Agile principles and values, making it a more collaborative effort. The book is a good read with rich elucidation of experiences and case studies of real-life engagements."

—Vishwesher Hegde, partner, PM Power Consulting

"Every organization has to deal with ownership changes in software projects. Hardly any organization, however, consciously plans for such transitions to be successful. The costs of poorly executed ownership transfers are huge and have a permanent impact on the organization and its business. Vinod Sankaranarayanan, in this book, draws upon his extensive experience in the software industry to build a framework and a step-by-step guide for us to use whenever we execute ownership transfers in our teams, organizations, or business-units. This book is a must-read and a must-have for IT project leaders and architects.

It is guaranteed to increase the chances of software ownership transfer success manifold."

—Vishy Ranganath, director of product development at Intuit

"I would suggest this book for anyone transferring any IT service. No matter how many transfers you have been a part of (specific service, acquisition, divestiture, etc.), there's always a gotcha you need to look out for, and this book makes you aware of most of them. Personally, I found the areas of Agile consideration to be most beneficial because this is so relevant for today's mature development organizations. It's obvious Vinod took his time to make this book so clear even the most non-techy of techies can understand it. It's easy to understand if you are a developer, manager, PM, or exec."

—Armando Morales, senior manager, Cisco IT

"Knowledge transition is one of the most important phases of a software development lifecycle. Vinod has very beautifully captured various aspects of the process and has given great suggestions so that other projects could benefit from it. The book has an interesting narrative based on real-life experiences and practical situations that keeps the reader engaged. While the book would be useful for any software

development methodology, the Agile way of doing the knowledge transition makes sure that both the parties actively pair during the transition to ensure the ownership transfer and not just to complete yet another milestone."

—Pravin Thakur, offshore development head, Thetrainline.com

"This book is a good reference for the ownership-transfer approach. It covers aspects related to both management and engineering practices. Vinod has very well explained his experience, to which we can relate."

—Anish Cheriyan, director-quality, Huawei Technologies, India

"Ownership transfer is not an everyday affair in project delivery. Neither is it so rare that people in the industry can wish it away. It is a crucial phase of projects and lays the foundation for a new beginning. At the same time, it is a perilous path as this phase brings with it many unknown unknowns. It is also dangerous because leaders are not always in control of situations given the multi-party involvement. Vinod brings out much needed attention and focus to an underrated topic through this book. The book provides details on both hard and soft aspects that play out during an ownership transfer. It also brings an interesting angle on using Agile principles for such activities. A must read for technologists who want to handle such transitions professionally."

—Padmanabhan Kalyanasundaram, head of the delivery excellence group at Mindtree

"We are starting to see the influence of Agile combined with digitization in almost every organization. Ownership of these initiatives is a key criterion to making Agile successful in today's fast-paced tech world. Vinod has captured the essence of ownership in delightful detail, including the challenges and pitfalls, which are illustrated very well with real-life examples. He brings in a refreshing approach to taking ownership utilizing the fundamental principles of Agile. A very useful book for transition managers and all members in the delivery organization."

—Jijo Olassa, CEO and cofounder, Verbat Technologies

"Project transitions are common in a multi-actor world. One of the key components of transition is knowledge transfer. Vinod provides a practitioner's multidimensional perspective on best practices to adopt covering technical and human-related aspects. It covers Lean Agile, the three bridges, and specific measurement metrics to quantify the efficacy of the process. These are based on real-life experiences from the trenches. If you are interested in quickly executing the insights in this book, you would find the Things to Know and Do sections extremely crisp and actionable. Overall it fills a much needed knowledge gap and can serve as an execution playbook for effective project transfers."

—Derick Jose, cofounder, Flutura

"Vinod has put together a compelling book that illustrates how *not* to be doing knowledge transfers. He has then married Agile philosophies to provide a completely different take on knowledge transfers, or *ownership transfers,* as he calls them. Since I had run a similar exercise, I could relate to a lot of the principles in the book. Coming from a product management background, I was particularly interested in the concept of 'continuous business'. The narrative-driven style ensures that the book is an easy read. The approaches given in this book will aid any IT organization as they execute their restructuring efforts or revamp their sourcing strategies."

—Linda Taylor, product manager,
thetrainline.com

"This book addresses a topic missing in current literature. It provides a valuable addition to the professional literature. In particular, the text is based upon the actual experiences of the author and his team. Numerous real-life examples are cited and bring the text a rich sense of practical advice. More important, the author has generalized the team experience and shown how this knowledge may be applied generally. While the book focuses primarily upon the issues faced by teams who use an Agile approach, most of the topics can be applied more globally. I have participated in ownership transfer, especially in the re-insourcing of IT services from Electronic Data Systems to the Blue Shield of California client team. Almost all of that development had been accomplished in a more traditional fashion. We would have benefited greatly from having a guidebook such as this to assist us."

—Daniel Scott, chief consultant,
LD Scott Consulting

"A thorough and interesting account of a complex handover. This book talks candidly about a topic that the industry prefers not to address openly. This unique insight provides lessons for anyone thinking about taking over or handing over the development and running of a system. Seeing where others have succeeded and failed gives real practical guidance on everything from the structuring of contracts to the running of the teams."

—Brett Ansley, CPO, VictoriaPlum.com

"Organizations constantly strive for operational excellence, which also translates into expensive initiatives in the form of business transformation projects. The most crucial aspect of these projects is not about going for the best solution but about ensuring transition into superior business processes with minimal disruption. Every servicing organization has a certain well-documented methodology for these transition activities. However, most of these methodologies suffer from near-sighted or one-sided risk mitigation strategies (for the vendor only) and often result in frustration and unhappiness on the client's part. Vinod is putting together a well-defined framework around this critical aspect of project delivery. This book will be a handy tool for novices as well as for seasoned professionals in providing a structured framework for transition with crucial business transformation projects. The emphasis on Agile methodology for knowledge transfer is very relevant right now as more and more customers are moving out of waterfall delivery expectations and are aiming for quicker payback from their investments. This book will help servicing vendors in delivering critical projects smoothly sans any operational risk or any cultural shock alike."

—Gaurav Mishra, enterprise data architect,
BMO Harris Bank

"Knowledge transition is important to any project. Vinod has shown how it can be made collaborative and effective while focusing on business continuity with his vast experience in ownership transfers. This book is useful for anyone attempting to move projects between teams located anywhere."

—Pramod Sadalage, coauthor of NoSQL
Distilled *and* Refactoring Databases:
Evolutionary Database Design

"Each IT application transfer project has its nuances and while there is a framework and best-practice guideline—the success or otherwise of a project is based on a clear understanding of its nuances and challenges, which would then pave the way for a well thought-out transfer program. Vinod has articulated well the need to strike a balance between leveraging such a standard framework and chalking out a fit-for-purpose program. Vinod has further brought this out through some options, such as, for instance, by outlining the efficacy of DevOps model, a model that seems to stand a much better chance of succeeding. IT application transfer projects often go off rails due to a short-term focus. Vinod has again described the importance of defining a scope that looks beyond the short-term milestones and considers the overall needs of the business—that is, the sustainability of the outcome of such transfer projects. A critical success factor for an IT application transfer project is the need for upfront and continuing trust among all stakeholders, and more importantly, an alignment on the mutual benefit of the project. Vinod has articulated well the people dynamics and how an 'upfront foundation of trust' could cut out wastes in the process and ensure that teams are working for a common purpose. Blatantly obvious, but Vinod has stressed the importance of this dynamic in a very compelling manner."

—Ravikumar MR, head of strategic operations- global markets, Allied World Assurance Company

"This book is not a theoretical dissertation on transferring ownership of a software project; it is a usable reference guide on how to set up a project to enable successful transition. Many of the ideas generated will lead to a healthy team environment and make for better all-around product delivery."

—Sameer Deans, delivery partner (principal consultant)

"This book is an excellent treatise on a highly critical subject, which is often not only taken for granted but also done incorrectly. The book is even more credible as Vinod has expounded the topic based on his extensive practical experience. IT products and solutions are highly valuable knowledge-based assets, and therefore it is imperative that the 'ownership' of knowledge transfer be done diligently. The approach propounded by Vinod, which is based on Agile principles, will certainly help in significantly reducing the disruptions and uncertainties not only during the transfer process, but more importantly, after the transfer as well."

—Sunil Mundra, Agile principal consultant, ThoughtWorks

"In a fast-paced world where knowledge transfer is often outstripped by the speed of business—with the next project often takes focus before the last project is fully embedded and sustainably maintained—Vinod provides much-needed practical insights that bring agility to this last mile of implementation."

—Betty Enyonam Kumahor, managing partner,
The Cobalt Partners

"This book breaks the misconception that software ownership transfer is confined to knowledge transfer. Through various anecdotes, Vinod takes us on a journey of the complete spectrum of software ownership transfer, discussing the technology issues, process issues, people issues, emotional issues, security issues, stakeholder expectations, etc. He also gives practical information on how to use Agile methodology in ownership transfer and how we measure progress. A good read if you are involved in any kind of software ownership transfer."

—Dattatri Salagame, COO – digital transformation and
enterprise solutions, Happiest Minds Technologies

"In an IT application knowledge-transition scenario, it is the ownership transfer that makes or breaks the day. This is seldom understood. Vinod has brought out this aspect with good anecdotes from other situations as well."

—Ramesh Ramakrishnan,
Tata Consultancy Services

"The book provides some excellent examples and food for thought to anyone who is considering or about to embark on the perilous journey that is software ownership transfer. Pulling from the concepts and principals of Agile development, as well as from the author's own broad experience, the reader will be well armed with a list of considerations to help them work through the handover process. A worthwhile read!"

—Cecile Diener, product manager, Equidelta

Software Ownership Transfer

This page intentionally left blank

Software Ownership Transfer

Evolving Knowledge Transfer for the Agile World

Vinod Sankaranarayanan

Addison-Wesley

Boston • Columbus • Indianapolis • New York • San Francisco
Amsterdam • Cape Town • Dubai • London • Madrid • Milan • Munich
Paris • Montreal • Toronto • Delhi • Mexico City • São Paulo • Sidney
Hong Kong • Seoul • Singapore • Taipei • Tokyo

Library of Congress Control Number: 2016942744

ISBN-13: 978-0-13-418101-1
ISBN-10: 0-13-418101-8

Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.

First printing: July 2016

Editor-in-Chief
Mark Taub

Acquisitions Editor
Chris Guzikowski

Managing Editor
Sandra Schroeder

Senior Project Editor
Lori Lyons

Production Manager
Ellora Sengupta

Copy Editor
Linda Morris

Indexer
Erika Millen

Proofreader
Jaikumar

Technical Reviewer
Sriram Narayan
Ramesh Ramakrishnan
Pramodkumar Sadalage
Pravin Kumar Thakur
Anish Cheriyan

Editorial Assistant
Olivia Basegio

Cover Designer
Chuti Prasertsith

Compositor
codeMantra

To my wife Vidya. She endures everything and endears everyone.

This page intentionally left blank

Contents

Register your copy of *Software Ownership Transfer* at informit.com for convenient access to downloads, updates, and corrections as they become available. To start the registration process, go to informit.com/register and log in or create an account. Enter the product ISBN 9780134181011 and click Submit. After the process is complete, you will find any available bonus content under "Registered Products."

Figure List

This page intentionally left blank

Preface

In 1990, C. K. Prahalad and Gary Hamel argued for organizations to focus on their core competence.[1] In many ways, this planted the seed for large enterprises to outsource their IT activities. It was around this time that IT service organizations bloomed and mushroomed. Many IT activities were first outsourced, then offshored. Organizations derived cost arbitrage, a 24-7 work cycle, apart from de-risking their information and knowledge assets.[2] The geographic spread of outsourcing provides business continuity guarantee. Twenty-five years later, almost every Fortune 500 company has either executed outsourcing and offshoring or, at a minimum, given considerable thought to its outsourcing strategies.

But times have changed. Today's world is replete with the paradigms of the Lean methodology, innovation, disruption, and of course, Agile. Indeed, we are witnessing another wave of Schumpeterian creative destruction.[3] Younger companies are leapfrogging established players to create products that are far more sophisticated and personalized. These changes are forcing organizations to rethink their sourcing strategies. Offshoring has given way to near-shoring (relocating to locations in neighboring countries), on-shoring (relocating to low-cost destinations within the same country), and, of course, insourcing (taking work back into the organization).[4]

At an organizational level, these sourcing strategies translate into ownership changes for application platforms. Applications and projects change hands as portfolios get realigned among the members of management. In this book, the focus is to explore how these ownership transfers can be executed with minimum risk and maximum efficiency. Ownership transfers are a consequence of organizational restructuring. Organizational restructuring is a vast subject in itself and perhaps requires another book. Hence, I have not dealt here on why these transfers occur.

As I wrote this book, the title underwent several changes. I started with *Agile Knowledge Transfer.* My then MD (managing director) at ThoughtWorks, Sunder Malyandi, suggested that this should be positioned at the decision-maker's level. Agile is serving a larger purpose, and that larger purpose is transferring ownership. I then changed the name to *Ownership Transfer,* but this title seemed confusing to many

1. https://hbr.org/1990/05/the-core-competence-of-the-corporation
2. https://www.flatworldsolutions.com/articles/top-ten-reasons-to-outsource.php
3. https://en.wikipedia.org/wiki/Creative_destruction
4. http://www.mckinsey.com/insights/business_technology/rebalancing_your_sourcing_strategy

people. It did not seem like an IT topic. After discussions with my editor and some of the book's reviewers, I then adopted *Software Ownership Transfer* as the title.

This is a perceptive book. I have drawn heavily on my experience in the IT industry to write this book. The literature in this space is quite sparse. Therefore, I have relied on interviews and focus group discussions with industry veterans to validate my hypotheses and fine-tune my suggestions.

During my interviews, it became apparent that what worked in my experience might not work for everyone else, at least right away. Although all of the interviewees agreed with the suggestions in the book, they felt constrained to practice them within their organization. Differences within management practices, business cultures, and sales organizations (in the case of IT service organizations) were all cited as impediments to adopt these measures. Interestingly, these are some of the same challenges in adopting the Agile methodology as well. Many interviewees told me that ownership transfers in their paradigm was a mix of development, maintenance, and operations. This industry witnesses large outsourcing deals running into hundreds of millions of dollars over several years.

The ideas suggested in this book may work well for Agile organizations, but they might be difficult to adopt within organizations that have deep divisions amongst development, maintenance, and operations streams. The ideas presented are more likely to work well in outcome-oriented teams as opposed to activity-oriented teams.[5] They may work well for teams that take a product view and teams that adopt a model in which development and operations work very closely with each other.

This is not meant to be an introductory book. This book assumes that the reader has experience in software delivery, appreciates Agile, and understands Lean methodology and Continuous Delivery. This book does not address every question about the transition process, nor is it meant for transitions in which the predominant focus is to rebadge incumbent team members to a new organization. Addressing all these topics would be like trying to boil the ocean. At the same time, there are some situations in the book that even a reader from the operations world can relate to. The book is not meant to be an exhaustive checklist of what to do in an ownership transfer, but rather one that analyzes many of the problems involved in ownership transfer.

The reader is urged to draw lessons from the many experiences and insights drawn in the chapters. I have discussed this topic in several forums and the most common concern of every practitioner is to keep morale high during the transition. It is the biggest challenge when undertaking this type of an exercise. It is the elephant in the room. However, keeping a team motivated is a leadership issue. The way each person solves this problem is a function of their personality, the culture within that group,

5. Sriram Narayan, Agile IT Organization Design http://www.agileorgdesign.com/p/table-of-contents-preface.html

and the policies in effect in that organization. Having said that, some of the stories in the book provide examples of ways team morale can be maintained.

I believe this subject has not been studied enough, and even with this book, we have barely gotten our feet wet. I fervently hope that organizations and groups will think beyond a ninety-day templated model for knowledge transfers. I will consider my purpose accomplished if organizations begin to look at ownership transfers holistically and plan for transitions over larger time spans with a deeper appreciation of the scope involved. I would be happy to hear any stories of organizations that realized their mistakes midway through a transfer and were successfully able to correct their course.

Who Can Make These Ideas Work?

This book is meant for medium-to-large organizations (from fifty to several thousand employees) who have significant investment in hardware, software, and personnel. Organizations and ISVs who find a need to execute sourcing transitions differently will find value from this book. IT service providers can use this book to engage effectively with their customers. Investors and business leaders may appreciate the insight that the cost of an ownership transfer is not limited to the cost of the effort expended in the actual transition. When a transition occurs, businesses risk a variety of costs, from the opportunity costs of failing to work on a business-critical project during the transition to the costs of service disruptions after the transfer occurs. This book offers ideas on mitigating these risks. In particular, the type of people who may benefit from some of the suggestions in this book are:

- Senior executives in IT organizations who decide on re-structuring and sourcing strategies
- Senior managers who are entrusted to drive organizational strategies
- Senior management at ISVs
- Executives in charge of procurement and vendor management
- Business heads who interface with IT groups
- Director or VP heading software delivery, product management, and engineering
- Senior Management of IT service providers
- Members of IT governance groups

- Process quality consultants, coaches, and SEPG group members
- Investors and board members deciding on diversification and consolidation strategies

Experience tells me that IT service providers have come to adopt Agile because their customers asked them to. The same will be required with ownership transfer. The ideas from this book have higher chances of success if they are driven from the customer organization. Throughout the book, I have provided many example scenarios and story narratives to make the concepts relatable and personal. Many of these examples are inspired by real-world incidents, but I have used fictitious names for companies and client personnel, and dramatized some incidents. I have retained ThoughtWorks's name because many of my experiences emerge out of my tenure at ThoughtWorks. At the same time, I have taken creative liberties with the incidents in order to better convey the message. Any resemblance to real-life situations is coincidental and unintentional.

These examples are not complete stories in that they do not provide a resolution to the problems; rather, they provide ways to deal with the problem at hand. In every instance where I have seen these approaches succeed, the implementers and their organizations were deeply invested. Their passionate belief and commitment to the cause (often more through deeds than words) ensured that the incumbents and the new members established working relationships and, as a group, became open to experiment and collaborate.

In a similar vein, I do not focus only on success stories. All change involves hard and soft aspects. Like an iceberg, the hard aspects are visible, but they comprise perhaps only ten percent of the efforts involved. The soft aspects are much larger and hidden beneath the surface. The undercurrents beneath the surface are what propels the iceberg forward. These soft aspects demand considerable leadership involvement. One of my aims has been to highlight challenges in running ownership transfers. I believe I have been quite candid in articulating a lot of our mistakes in running these programs. As they say, success inspires, but failure teaches. I am positive that many of the examples illustrated will resonate with your own experiences and the insights presented will aid you in solving future problems.

How to Read This Book

The classic struggle of most authors of this genre is to keep the reader engaged while at the same time pass on knowledge. Hence I chose the narrative format. Throughout this book I used the ownership transfer that occurred at EuroT as a preeminent storyline to discuss various themes on the subject. Chapters 1 and 4 lay out the basis

of the EuroT project. Chapters 5 through 12 pick out specific dimensions of this topic and discuss various nuggets of thought.

At the same time, I also included several other stories, such as that of Arival systems, which provide a very different perspective—that of someone taking over an application. Chapters 12 through 15 step away from the storyline to discuss general guidelines on approaches to sizing, measurement, and addressing the complexities arising out of these exercises.

I urge the reader to go over each chapter in the order laid out. For the hurried reader, I have listed a summary section of "Things to Know and Do" at the end of each chapter. For those looking for a lay of the land, I have included the following mind-map. The mind map shown in Figure 0.1 provides a holistic view of all the dimensions covered around ownership transfer. Each of the level 1 subnodes broadly translates into a chapter within the book.

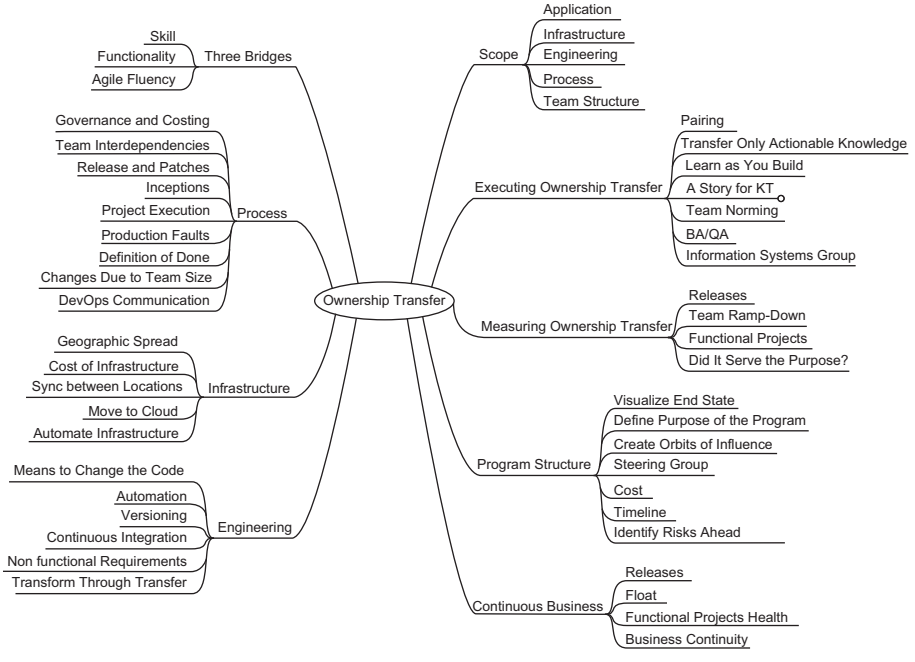

Figure 0.1 *Ownership transfer—mind map*

Register your copy of *Software Ownership Transfer* at informit.com for convenient access to downloads, updates, and corrections as they become available. To start the registration process, go to informit.com/register and log in or create an account. Enter the product ISBN 9780134181011 and click Submit. After the process is complete, you will find any available bonus content under "Registered Products."

This page intentionally left blank

Acknowledgments

This book would not have come to life if not for the constant source of motivation from Pravin Thakur, the offshore head for thetrainline.com. He has seen this book evolve from a presentation to a rough script to a complete manuscript. In each stage, he was present to provide his ideas, feedback, and guidance.

David Jack, Group CIO MetaPack, germinated the idea of looking at this discipline as a best practice. That conversation was the seed that goaded me to delve deeper into the world of knowledge transfers.

Pramod Sadalage, renowned database expert and author of many books, put me in touch with Bernard Goodwin at Addison-Wesley, who became my editor until his retirement at the end of 2014. Christopher Guzikowski took over from there and brought my book to its conclusion.

My development editor Michelle Housley had been my constant point of contact starting with the contracting process with Pearson, through the proposal reviews, peer reviews and finally providing me with her valuable development edit insights.

This book has achieved its final form thanks to the copy editing and compilation efforts of Ellora Sengupta and Linda Morris. Their eye for detail continues to amaze me.

I need to thank every friend of mine who readily agreed to participate in our focus group discussions. Those discussions helped validate my hypothesis and suggestions on this topic. Many of you readily shared your personal experiences as well. Some of those have found their way into this book. Thank you Sudheer, Sreeja, Rahul, Sriram, Preethi, and Sudeep.

My reviewers played a critical role in chiseling the theme and structure it to its current form. Their input gave me the opportunity to add several chapters to the book as well. Many thanks to Pramod Sadalage, Babuji Abraham, Ramesh Ramakrishnan, Sriram Narayan, Anish Cheriyan, Vishewshwar Hegde, Sriram Narayanan, Vishy Ranganath, and Ramesh Dorairaj. And of course, many thanks to all those who read over the manuscript and provided endorsements.

I extend my gratitude to the ThoughtWorks operations team, who approved my vacations to work on the manuscript. I also thank Suresh Kalarikkal, who provided legal advice on various aspects on the book.

Ashay Saxena provided access to the Indian Institute of Management, Bangalore (IIMB) campus. The relaxed environs of the sprawling campus blew away my writer's blocks.

I must thank Suchita, Nikhita, and Akshay for sharing their personal experiences with transitions. Their stories have helped me enrich the manuscript.

And finally, my wonderful wife Vidya, who has happily accommodated my cooped-up existence weekend after weekend over several months. And our two handfuls of joy, Dhruv and Dyuthi; our rays of sunshine. They make every dull day a delight.

About the Author

Vinod Sankaranarayanan, a Project Management Consultant at ThoughtWorks, has advised many organizations in the Finance, Travel, Retail, and Healthcare space. Along the way he has handled several large project delivery responsibilities. He has witnessed and driven several sourcing transitions for his clients.

Vinod started his professional career in the pre-Agile era and later adapted to the Agile model of working. This has enabled him to empathize with and provide practical suggestions to clients who often find themselves making Agile transitions and sourcing transitions at the same time.

Vinod has essayed the roles of a business analyst, tester, pre-sales consultant, program manager, account manager, Agile coach, and practice leader as part of his long IT Journey.

Before ThoughtWorks, Vinod worked at Mindtree, a well-known IT services organization. At Mindtree he was one of the founding members of the Business Analysts Council and the Agile Council, focusing on creating organizational capacity to win and execute consulting as well as Agile projects.

He lives in Bangalore with his wife Vidya and their children Dhruv and Dyuthi. An occasional blogger and speaker at conferences, his writings, talks, and contact information are available at ownershiptransfer.in

This page intentionally left blank

Introduction

In December 2012, Mason, our development director, flew down to Bangalore. Every time he visited Bangalore, we always kicked off with a Leadership Team meeting. This time, though, Mason had a unique agenda. After having worked together for over five years, the time had come for us to transfer responsibility over to the EuroT team at the headquarters in London. We needed to put a plan in place to transfer work that we traditionally did out of Bangalore over to London. There were many ramifications to this transfer: London needed to pick up more responsibility. Its team size would grow. The team in Bangalore would ramp down. We were not clear if we would have a team in Bangalore any longer or not. We were not sure what and how much could be transferred. We had certainly not assessed the team's reaction to this new course. Not many in London knew of the decision. We essentially needed to chart a course for our ship, but the destination was unclear.

EuroT started off as a travel agency in the late 1980s. It is now a large online travel company specializing in selling bus and train tickets for a huge commuter base in Europe. EuroT's specialty is to bring together schedules and connections from disparate bus and train services across Europe. After the advent of the Internet, EuroT employees realized that travel booking would change forever. In the late 1990s, EuroT was one of the first companies to set up an online reservation system. Since then, they expanded to include payments, personalization, and a slew of other features in their application. Today, they are the preeminent online travel company in Europe. Their website and mobile applications are some of the most popular in Europe. Thousands of commuters use their applications for their daily commutes. More than 20,000 tickets are sold every day on EuroT. The platform witnesses more than eight billion euros in ticket sales every year. EuroT's technology platform is so robust that many other bus and train companies use its engine to power their own ecommerce websites. Thus, EuroT caters both to the business segment and to the consumer market.

ThoughtWorks is a well-known name in the world of Agile software development. They apply avant-garde methods to designing, creating, and delivering software. Spread over 30 offices in 12 countries, their more than 3,500 consultants help organizations large and small to deliver business-enabling software. EuroT had requested that ThoughtWorks review their software design and code.

This relationship, which started as a one-off task, eventually translated into one of the longest and strongest partnerships both organizations ever had, one which ran for many years. As the software platform matured and the business gained robustness and growth, it was time to level down the engagement. Although many factors were behind the decision, the primary driver was EuroT's need to reduce capital expenditure.

That week in 2012 was a busy week for the EuroT Leadership Team. We had trouble figuring out where to start. It seemed a bit like jumbled tape. If you can't get hold of one end, you cannot begin to straighten it out. Systems have a way of becoming complicated. We had created many processes over the five years of partnership—inceptions, releases, defect triages, and nonfunctional testing (or NFT, as we called it). Each stakeholder had a different view on what was crucial. For the development manager, story development was preeminent. The release manager considered defect triage and the release checklist critical. For the test manager, regression and nonfunctional testing were important. Delivery metrics were crucial to the delivery manager.

Some of us realized our current roles may no longer be relevant after the transfer. Discussing such a transformation objectively, especially when you realize that the outcome may make your role redundant, calls for tremendous maturity.

This was a pivotal point for the account. We clearly had to bring everyone up to speed with the direction being taken. We also needed to ensure that the usual way of thinking changed. Decisions on projects, modules, components, and even "business as usual" activities needed to be made with the transfer in mind. Everyone needed to ask additional questions when making decisions. What does it mean for the transfer? Will the Bangalore team be doing this the next time? If not, does someone in London know what needs to be done?

Jason Gots, editor at BigThink, says that there were five big moments in the history of knowledge transfer.[1] The first milestone was when we began using smoke signals. The ancient Chinese used them to communicate military tactics. The Roman College of Cardinals uses the smoke signal to convey the election of a new Pope, even to this day.

The carrier pigeon has a checkered history dating back to Ancient Rome. Their reliability prompted the Pigeon Post of the Great Barrier Island in 1897, a postal service that relied on the pigeons flying across islands to deliver messages. They were used heavily during World Wars I and II.

The printing press allowed further widespread distribution and retention of knowledge. During the Renaissance, printing costs decreased, which allowed for an increase in publishing and sharing knowledge. Although television and radio

1. http://bigthink.com/think-tank/5-big-moments-in-the-history-of-knowledge-transfer

are far more dynamic media, they do not store information like printed materials. The printed word allows you to retain information and revisit it as need be.

The telegraph was another watershed event in the era of communication. It allowed simultaneity across the world.

However, all of the previous mediums supported communication that is one step removed. Many of them did not enable synchronous communication, allowing for simultaneity on either side. The Internet has helped break barriers and build bridges. It combines the benefits of all the previous inventions and provides a truly empowered medium for sharing knowledge. Whether through social networks, online classroom sessions, or textual databases, the compendium of information harnessed over the Internet is overwhelming. Although communicating online can never replace being there to share a moment, it can do the next best thing in most cases. Open a champagne bottle on both sides, raise a toast, and clink the LED.

Over the next few chapters, I discuss what I learned over the ensuing months not just from my experience with EuroT, but also from others' experience in similar exercises in other organizations as well. The chapters cover eight major themes: executing ownership transfer, being Agile, ownership, engineering, process, infrastructure, culture, and continuous business. The last chapters are dedicated to how to apply this knowledge to future transfers.

This page intentionally left blank

Chapter 1

The Challenge with Knowledge Transfers

Every IT product goes through a knowledge transfer (KT) phase at some point in its lifetime. Quite often it is immediately after development is completed and maintenance has taken over. Sometimes other reasons can trigger a knowledge transition. Outsourcing, an incumbent's poor performance, or business landscape changes can trigger a need to hand over an IT product.

While going through this phase, enterprises often do not have a predictable way of transitioning. In outsourcing scenarios, one relies on the expertise provided by the vendor taking over the application. In cases of restructuring or insourcing, organizations might not have a ready reckoner to look at how these transitions can be handled.

Most IT service organizations use a readymade template for knowledge transfers. The processes in these templates rely on a set methodology of analysis, shadowing, reverse shadowing, and steady state. Some organizations also talk about a stabilization phase. Contracts are signed up with timelines of three months, often followed by a stabilization phase. However, the delivery teams may face inordinate challenges:

- **Challenge 1:** The most obvious challenge is that there is a lot to do within a short span of time. Apart from everything else, assembling the team, knowing the lay of the land, and putting a plan in place will shave off a few weeks from the schedule.
- **Challenge 2:** The next challenge is to get the customer and incumbent to agree to the plan. This may take longer than you expect. In fact, it can take a month or more for people from three different organizations to even get into a working rhythm. It takes time for momentum to build up.

- **Challenge 3:** People soon realize they are falling behind schedule. With strict timelines, there is a tendency to cut corners, and quality begins to suffer.
- **Challenge 4:** Most well-laid plans bring out unknowns. Strict timelines often do not allow for time to unearth the unknowns.

A tactical challenge faced by the new team is getting all the right access to all systems within this timeframe. Assembling members is a crucial step before getting the access rights. Only when the members are assembled can we tell which individual needs what access. After that, we need to identify all of the ports, firewalls, and authorizations required to execute each member's work.

To say that we need to list all the necessary ports, firewalls, and authorizations in one place and then get IS resources to implement the access rights makes things sound simple. The process is rarely that straightforward. In reality, even members who have worked in the group for a long time might not know all the systems to which they require access.

In one of my recent engagements, in which I was working from the client's location, we needed to obtain certain access rights and a hard token to allow members to work remotely. The client manager put us in touch with an old vendor who had remote access. This member had obtained access more than two years back. He came by to help and soon realized that the customer processes had changed. We had to get four different tickets created for different levels of approval, and each ticket required approval from different people. In addition, the tickets expired if approval did not come within two days. It took two to three months for some team members to obtain this access. In another organization, the process of granting administrator privileges on developer work stations had its own unique workflow, and in one extreme case, it took us a full month to grant a developer administrator rights.

Additionally, it's not easy to gauge the quality of the transition. How do we gauge whether the knowledge sessions are really effective? How do we really know if we have understood everything? Any incumbent employee who has been working and supporting the system for a long time may well have established a few nifty things, such as little scripts or utilities for monitoring or promoting a piece of code, to make their jobs easier. Such things may not have been documented, but would aid in a lot of debugging issues and expediting delivery. Further, we may not see enough issues crop up to demonstrate all the measures used for maintenance and enhancements over a short span of time, whether through workshops or in the two to three weeks of shadowing and reverse shadowing.

In all cases, a stabilization phase continues over long periods. In my experience, and those of several colleagues who have taken over large enterprise platforms, it may take almost a year before you really feel in control and are operating at a level of optimized efficiency.

Post Takeover

The period after takeover always shows a dip in production availability and reliability. Teams go through long hours of stress as they battle production tickets and unknown areas of the application. Time taken to fix issues increases. Service organizations recognize this. Many contracts specifically ask for a waiver of penalty (or a holiday period) in the first few months on steady state. By the time teams feel in control, there is only a sigh of relief rather than joy. This period heavily tests the business and the relationships between organizations and amongst individuals. Some of my colleagues from ISVs have mentioned that the team literally slogs it out in the first year after takeover. Although many organizations are certified at SEI CMM Level 5, when a team hits a production snag, those certificates don't carry much weight. People need IT heroes in the team to pull things through. Dependencies on folks who are technically competent, can think on their feet, and can manage time pressures increase that much more. This has become de rigueur in today's knowledge transfer world.

Unfortunately, many times the outcome is not positive. Business impact can be huge when product performance is compromised. Older team members are called back, perhaps poached from other organizations, as desperate measures. Sometimes entire teams are replaced, creating, in a sense, a transition within a transition with far greater urgency. For many people, this becomes career limiting. Many senior level executives have lost their jobs when transitions have failed.

ITIL versus DevOps

Over the last few years, we have witnessed the industrialization of IT operations. ITIL[a] (Information Technology Infrastructure Library), a set of practices for IT service management (ITSM), has matured over the last five years. We now see the advent of automation in this space. Companies such as IPSoft[b] are using robots and artificial intelligence to support IT and business operations. These will continue to be relevant for large platforms running on COTS (commercial off-the-shelf) software implementations with limited customizations. In fact, with industrialization, we are seeing transitions mature and become more efficient in the COTS space.

Our world is hurtling relentlessly into a space where differentiated experiences and a quick time to market are becoming keys to survival. At the

a. http://searchdatacenter.techtarget.com/definition/ITIL
b. http://www.ipsoft.com/

(*continued*)

moment, there are quite a few differences between ITIL and DevOps. In order to aid this transition, I believe the ITIL paradigm will change to include key principles from the DevOps world:

1. DevOps takes a holistic view and stresses on the performance of the entire system. ITIL looks to break down the whole into phases of activities.
2. DevOps looks for a cross-disciplinary approach and stresses on outcome orientation. In the traditional ITIL set-up, there are clear compartments between Development and Operations. Even within operations, there are clear distinctions between Release Engineers, DBAs, Network Engineers, and Operations Staff.
3. DevOps tries to amplify feedback and continuously learn. The ITIL ethos follows planning control and documentation. ITIL tries to bring consistency by creating processes and checklists.
4. Both ITIL and DevOps use automation. In the case of DevOps, the focus is on automating routine tasks. With ITIL, artificial intelligence is being used to mimic human behavior.

To put it succinctly, the future of ITIL is one in which robots take over human activities. The ITIL paradigm (as does the Industrialization principle) looks to remove the uncertainties brought in by human interactions. In the DevOps world, on the other hand, the future looks to automate verifications done on the system by team members. DevOps and Agile celebrate the social angle of software development and maintenance.

Contracting

Enterprises are starting to recognize the need for longevity of key members. Today, customer organizations seek to get the names of key members put in the contract. Exit clauses in contracts are getting longer and longer. IT set-ups are learning the need to ensure continuity and cooperation in the case of handovers. There is an appreciation within enterprises that team transitions are becoming the norm rather than the exception. More than seven pages of a contract can be dedicated to engagement termination. These contracts touch upon various aspects from setting up transition managers and knowledge transfers to the rebadging of vendor personnel. Some contracts have even begun to require that client nominees be able to work from the vendor offices for purposes of transition. One must commend the proactive think-through on this subject at the contracting phase itself. The irony, though, is that organizations are trying to secure cooperation through contract negotiation.

Timing

A rush to complete transitions can have a detrimental effect. Let's take the example of banks. Bank activities are generally seasonal. Year-ends are crucial periods. Transitions in today's banks are planned so that teams do not "mess up" this crucial time. In effect, there is a rush to complete the transition before year-end activities begin. But what is often overlooked is that the new team is suddenly facing a set of crucial activities that they never had experience with. It's a bit like fielding the national squad for the World Cup with a set of players who have never played in a high profile and high pressure arena. No team has won the World Cup while fielding an entire squad of greenhorns. Instead, winning teams always have a judicious mix of youth and experience. Similarly, in the case of transitions, is it not better to have the new team pair up with the incumbent on an activity as critical as this, rather than relying on the new team's untested skills and capabilities to pull through?

Scope

Today's knowledge transfer focuses on taking over the support and maintenance of an application. However, no focus is given on how teams should work to enhance the application. Development is different from support. The exercise of inception to working with business on showcasing stories is unique to every organization. In three months, which is about the duration of today's transition agreements, many teams don't get an opportunity to partner with the incumbent and deliver an enhancement. Imagine a scenario when the old team is no longer around and the new team is tasked with an enhancement. They meet the business folks and they are told "that's not how we do it here," or worse, "that's not how the old team used to run this." That can make the situation quite awkward, and things can quickly decline from there.

Today's Transition Program Manager

Suresh is an experienced IT program manager working for a large service organization. He has been working in the IT industry for over fifteen years. He has led several successful transitions in his career. When his team contracts with a new account, they identify large development projects expected ahead of time. Then, they request the incumbent to complete these projects. Because Suresh's team will have their hands full taking on the maintenance aspects of the existing platform, they don't have time to focus on the new projects. Suresh realizes that this goes against the grain of practical wisdom. The new projects drive innovation. New projects will have more maintenance and enhancement challenges after they go live. Yet his team continues to focus on the matured systems even as work in progress (WIP) streams do not attract any focus.

Organizations are particularly keen to understand how to transfer over projects that have been developed in an Agile fashion into maintenance mode, and they are asking this specific question to the service organizations they hire. No one has really broached this topic in-depth. This is primarily because development phases have picked up Agile models, whereas, in many instances, the maintenance and operations teams work with older methods. Consider the case of a colleague who was coming out of a large-scale transition of a development project into operations. In this case, the biggest challenge he faced was that every time the development phase needed to handover to operations, they began hitting roadblocks. Operations would say that adequate documentation was not present to take over the application. The development team would say that a working product is more important than comprehensive documentation.[1] Both development and operations view things from different positions on Agile adoption. In an ideal world, the operations team would have been involved with the project from the requirements and development phase. On the flip side, the expectations of the operations team would have been factored into every story played by the development team. A few organizations are beginning to restructure this into a DevOps model.[2] The DevOps model focuses on both development and operations with the intent to provide continuous delivery. To put it in another way, in this paradigm, development and IT operations happen continuously and seamlessly. However, the vast majority of organizations still do not operate with this philosophy.

True Cost of Transfer

A related issue to knowledge transfer is its limitations on business advancement. Many organizations have a tendency to pause project development while knowledge transfer is in progress. It is understood that projects may be paused for about three months. However, in reality, the impact is much longer. The entire stabilization period slows down delivery. New features and functionality may not reach the market on time. This adversely impacts the competitiveness of the organization, which may be the very reason why a transfer was triggered in the first place. This is another phase in which the relationship between business and IT is tested immensely. The confidence in IT decreases within the organization. Apart from the obvious issues of delivery slow-down, IT members may fail to have the right levels of context and domain understanding to engage with business members effectively. This leaves business members spending more time with the added task of detailing out functionalities and expectations for the IT members. Business members will also

1. http://agilemanifesto.org/
2. https://en.wikipedia.org/wiki/DevOps

need to acutely focus on testing out the new features with the new team. However, none of the knowledge transfer metrics currently track these impacts.

For too long, we have focused on the operational efficiency of IT projects as opposed to their effectiveness. The project charter goes through deliberate discussions on the purpose of the project. Most projects quantify the expense and the expected benefits of doing the project. A business case is put out to justify the return on investment (ROI). After this has been approved, everyone focuses on staying within the cost, timeline, and scope of the project. We often do not step back to see whether the real purpose of the project is still relevant or whether there might be other opportunities along the way. The metrics on knowledge transfer continue to focus on operational aspects such as how much has been transferred, the cost incurred so far, and the team's ability to stick to their deadlines. We don't try to determine how well things have been transferred. So although we can have a lot of "completed" tick marks, true internalization hardly ever occurs. We get what we measure. Unfortunately, in this case, we aren't measuring the right stuff.

The inherent costs of a transfer run much deeper than the time and effort spent directly on the activities. Costs of business disruption after the handover or during the stabilization phase can be debilitating. However, these costs may not be directly attributed to the knowledge transfer, often because the project may have been "completed" by then. The opportunity costs of failing to put functionalities in the market can be quite heavy. Needless to say, the ecommerce world is rapidly evolving. Competition exists in every sphere and customers are getting savvy and demanding. One can ill-afford to pause innovation that customers look for. Such costs are rarely accounted for within the metrics tracked on a knowledge transfer project. These costs can be several times higher than the direct cost of personnel in the project.

Agile software development has seen substantial adoption among organizations. Agile and Lean delivery focus on high-engaging, non-fussy models to deliver software. Most importantly, the cultural shift in the Agile model requires a substantial people-centric approach. Today, many software platforms are developed using Agile methodologies. More importantly, there are several benefits in utilizing Agile philosophies when undertaking knowledge transfer programs. Agile focuses substantially on business outcome versus operational efficiency, on quicker outcomes along with quicker material feedback. Agile also does not focus on "big bang" releases. Instead, it encourages us to release in short cycles, allowing us to refine our processes and get feedback. Adopting a similar approach to knowledge transfer would allow us to reap the very same benefits. Smaller milestones allow the team to get quick wins, increasing confidence all around. It also allows us to learn and adapt. This is even more crucial in a project whose execution will be nebulous. These projects will witness heavy undercurrents given teams from different groups are involved. This is all the more reason to give such projects more time to think big while starting small.

Practical World

Many of my friends and colleagues work in the IT industry. Many work in service organizations and quite a few work in IT groups of large corporations. We have had several discussions on the concept of ownership and with every conversation, a new challenge arises. The real world is far murkier than the pristine idealism of cooperation and partnership.

Competing Priorities

These days, customers commonly ask for key project personnel to be mentioned in the contract. However, a "seasoned" service organization manager would never put his stars on that list. In fact, delivery managers take pride in ensuring that the real performers of the team do not get caught into the contract trap. The moment the incumbent team's management realizes that they are losing business, they will focus all their efforts on finding new pastures where better business opportunities may exist. This is a classic case of competing priorities. In an Agile paradigm, teams need to spend time in handing over ownership, but if the parent organization of the incumbent has a conflicting interest and seeks to pull out key members, this will never occur. There are no easy answers to this practical problem, and most often, the issue cannot be resolved using contracts. Customer organizations need to think from the viewpoint of the incumbent vendor and understand what would appeal to them and keep them on-board. In our case, the EuroT CIO himself suggested that we use this exercise as a way to build a center of excellence on knowledge transfer. That sounded ambitious to many of us. Some of us had already experienced knowledge transfers in prior engagements, but the opportunity to look upon this as a discipline in itself and elevate it to a higher order added a new level of enthusiasm. Since then, we have carried over what we have learned as we moved on to other engagements.

Rebadging

A transition often means hiring some of the employees from the incumbent organization. The industry-accepted term for this is *rebadging*. Rebadging derisks the program and application to a great extent. However, rebadging can also have its pitfalls. Employees may not be ready for a rebadge. The good ones may find better options and move on. In most cases, employees may not be happy with a move from a customer organization to a vendor organization, for example, or across vendors. Personal equations and egos get in the way. Eventually these people quit, taking their knowledge with them.

However, rebadging has become an accepted norm. Many of my colleagues vouch for the efficacy of this practice. In reality, what these organizations are doing is

buying more time for their intended team to build context. The members who have been rebadged will go through a completely different transition—a new organization, a new culture, and perhaps a different career path. Many may have transferred to the incumbent because it seemed to be the safest option at that time; but most will reconsider.

A client with an Agile organizational culture would have a fairly sound idea of the real performers from the incumbent. The incumbent management runs the risk of losing some of these employees if the client or the new vendor decides to make offers. These types of situations can become lose-lose very quickly. Therefore, it's in everyone's interest to sit down and make a commitment to make the transfer successful.

Story of a Rebadged Employee

Sunita worked at the Mumbai center of MegaTos, the technology division of Mega, Inc., a global leader in the banking world. She was a young IT engineer with about two-and-a-half years of experience. She was one of those team members who jumps in and solves problems with verve and vigor. She enthusiastically volunteered for all activities, be it testing, analyzing logs, or comparing features across environments. She was part of a 2,500 member IT group. In November 2010, they were informed that their entire division was to be merged with Vettai, Inc., a large India-based services provider employing over a hundred thousand people. The merger was to happen within three months. By January 2011, all 2,500 members would be employees of Vettai, Inc.

In the beginning, everything was the same. Sunita went through the induction processes from Vettai, Inc. Like everyone else in her IT group, she got the joining kit and had a sense of being made welcome. Even after January, the members of the IT group continued to receive their salaries in the MegaTos format with their MegaTos ID. Their Vettai email IDs had not been set-up. Not that it mattered to Sunita. The location was the same, the facility managements were the same and nothing really changed until May of 2011.

Sunita noticed that the fifteen members of senior management from MegaTos had resigned. The CEO of MegaTos had left in January. Given her profile and the nature of her work, this did not make any direct difference to her. However, other things began to bother her. MegaTos had implemented a product from DhanTech, Inc. Of late, the consultants from DhanTech had become less responsive to her. DhanTech had removed certain access privileges she had enjoyed earlier, making her feel handicapped. She was no longer their client, but a partner vendor.

(*continued*)

In the first week of June of 2011, as the employees were walking in to work, security replaced their MegaTos badges with Vettai badges. This was a full five months after the transition was supposed to be completed. The merger had finally become a reality in their world.

At Vettai, Sunita started to feel like she had to prove herself all over again. Vettai, Inc. was a 150,000-employee behemoth. Rewards and recognition systems were vastly different from what she had known. At MegaTos, the employees received a bonus based on the company's performance. At Vettai, their bonus was driven based on the performance of many other projects and accounts, teams that Sunita had no clue about. At MegaTos, she could work from home when required. All she needed to do was inform everyone by email. At Vettai, she had to request it in advance and her request had to be approved by a manager.

Sunita was married at the end of 2011. She wanted a transfer to Pune, a city 150 kilometers from Mumbai, since her husband worked out of that city. Fortunately, Vettai, Inc. had a thriving work center in Pune. She put in a request and was fortunate enough to get a work allocation to the Pune office. However, her first day at Pune office in January 2012 proved to be a shock. Outside of the Mumbai office, she was not recognized as an employee anywhere else within the Vettai internal systems. Her access card did not work. She had to prove to security that she was actually an employee. She learned that the 2,500 employees retained from MegaTos had been tagged as an XT044 group whereas the regular Vettai employees were XT01. She required a company bus for her commute. But with her XT044 tag, she could not take advantage of that benefit. Whenever she needed to work late into the night, she had to use the cab facility, but she did not have access to raise the request.

A year after the announcement of the transition, Sunita had transitioned out of MegaTos, but she had not been transitioned into Vettai, Inc. For the next five years, former MegaTos employees were not allowed to work on any other project within Vettai, Inc. Within three years, the formerly 2,500 strong MegaTos group had shrunk to 150 members. Eventually, Vettai, Inc. closed down the Mumbai office, the building that had housed the erstwhile MegaTos office.

The Evolving Nature of the Program

Areker Bank is one of the largest banks in the U.S. Their current banking platform, named Lothar, has been running for the last thirty years. Over time, their customer demographics began to change. Customers had become more connected, empowered, and demanding. The bank implemented a next-generation banking product to serve this new-age customer—Banker, a product of Sofa Technologies, one of the largest software companies in the world. This program was christened as Nerup.

Areker planned to utilize Agile methodologies for the implementation. They took over direct responsibility for implementing Banker from Sofa. Areker engaged LeanAgile, Inc. in the transition program. The plan was for LeanAgile to take over the Banker product implementation from Sofa within a span of six months, but as the engagement progressed, they realized that Sofa Technologies brought deep product knowledge, something that neither Areker nor LeanAgile would be able to pick up within six months. Because this was still a fairly new product, only personnel within Sofa technologies had enough knowledge of the product to effectively implement it. Such skills were not available elsewhere in the market.

As this realization dawned on the parties, they extended their engagement with Sofa Technologies. The teams have now been working on Banker implementation for more than two years. Even to this day, the implementation team is largely comprised of Sofa engineers. They were able to change course because at some level they were following the Lean principle to defer commitment.[3] This principle states that decisions must be frozen as late as possible in the process, especially those decisions that become irreversible. If they had made up their minds to take over and sealed down contracts and timelines accordingly, the banking system would have crashed, and most likely Areker Bank would have gone out of business.

Politics

Most IT organizations take a radically different approach than what Areker Bank chose. Knowledge transfer often begins with a surreptitious agreement between the customer and the new vendor. This period sees heavy negotiations. The client is keen to get good rates and a robust set of SLAs (service level agreements).[4] The vendor is keen to win the business. All of the negotiations are centered around rates, the SLAs, and how soon one can take over. In effect, all of the negotiations are about improving

3. www.allaboutagile.com/7-key-principles-of-lean-software-development-2/

4. http://searchitchannel.techtarget.com/definition/service-level-agreement

IT efficiency. There is little to no focus on business outcome. The vendor uses their prior knowledge with similar engagements to negotiate. The client uses their historical experience with the incumbent to negotiate. All through the process, both IT parties overlook the purpose of the product.

Unfortunately, these discussions and the negotiated contract are what drive team behavior. Given the need for discretion, both the client and vendor make many assumptions about the product and the environment in which it will be used. The vendor is most interested in winning the account and making a sustained profit on the account. The IT organization is looking to drive down costs and maintain SLAs. Teams are trained to show their SLAs are within limits. If the product fails, it could be marketing's fault or the business team's mistake because they did not know what their customer wanted. However, the IT organization and the vendor are happy if they are within their SLAs.

After the contract is signed, the onus is upon the incumbent. The obvious reflex is to resist and turn hostile. Even if the technical team on the ground is open to help, the incumbent's management might make it as difficult as possible. The incumbent account manager might be worried about his job. Last-ditch efforts are made to win back confidence. A number of concessions are provided. However, given that the client and the new vendor have signed what is considered a rock-solid contract, there is no wiggle room for the incumbent. When this becomes obvious, the incumbent management may lose interest. They might try to pull out their best members to focus on "rising accounts." And so, all of the team's and management's energy goes into getting everyone to interact in a productive manner. Driving this behavior creates a lot of waste in the system. Nine times out of ten, the clauses mentioned in the contract will fall through. The SLAs become unachievable. Many of the original assumptions turn out to be wrong. The product suffers.

Losing Incumbent Members – A Different Perspective

The Banker Product skill is very much in demand. A number of Sofa engineers have quit the Nerup program. This poses a risk to the transition. However, the Areker management took a unique perspective on this. They believe that with their in-house team and the LeanAgile team going through the transition, they are actually better placed and more secure for the future. In effect, Areker Bank believed that the incumbent personnel would, in any case, have moved on at some point.

Things to Know and Do

- Program schedules can be unrealistically short given the operational challenges to execute such programs.
- The true cost of transfer includes the opportunity cost of missed projects and revenue loss due to increased service disruptions.
- The focus on transitions is often on efficiency and not effectiveness. There is a greater focus on activity over outcomes.
- Scope may be focused only on maintenance and not enhancements. This gives no opportunity to prioritize what needs to be transitioned and what doesn't.
- Politics may creep in between the three groups: IT management, the incumbent team, and the new team.
- The program may evolve differently from what was envisioned. Watertight contracts do not provide wiggle room to course-correct the transition.

This page intentionally left blank

Index

D

E

F

G

H

I

J-K

S

T

U

V

W-X-Y-Z