

Praise for *Lean Integration*

“What the authors have set out here is a philosophy built on best practices from both the fields of manufacturing and software development, but they do so with examples that bring the material alive, come from real life, and offer specific, measurable actions and practical alternatives. This work is fantastic, not just from a technical standpoint; it has a maturity that’s vacant from other works, an understanding of internal business politics and human resources concerns, all the while wrapped in solid management principles and practices.”

Kevin P. Davis, Senior Technical Architect

“Technology is a key enabler within any industry and a key success measure is the ‘alignment’ between business and information technology. Schmidt and Lyle provide practical advice for a fundamental shift in thinking, from IT as an internal services function to IT as an integral part of a company’s strategy, creating value for customers. IT internal and external service providers have to operate as one management team. *Lean Integration* presents compelling examples of how integration teams play a role in leadership, strategic planning, and IT governance as some of the critical factors in achieving organizational alignment.”

*Zahid Afzal, Executive Vice President/Chief Information Officer,
Huntington National Bank*

“In today’s world, enterprises struggle with increasing global competition, the need for speed to market, and the ability for IT to enable the strategic intent of the business. One of the core tenets of lean that many integration professionals lose sight of is the need to put the customer first. This book serves as a reminder to our fiduciary responsibility to leverage IT as a competitive tool for planning and execution.”

James McGovern, Enterprise Architect, The Hartford

“This book should help the IT executive and practitioner, alike, align on goals and objectives that drive long-term value to their enterprise. The Integration Competency Center can drive as much or more value for the IT department than any other capital investment it will make in the next decade.”

Clark T. Becker, Former SVP and CTO, Best Buy Co., Inc.

“In this highly communicative world, one filled with a high degree of turbulence and uncertainty, the one key that will separate successful businesses from the rest is their ability to be agile and wield just-in-time, focused, trustworthy information. I am extremely pleased to see that John and David have written on such an important topic.”

Mark Albala, President, InfoSight Partners, LLC

“John Schmidt and David Lyle have written an important book with a new perspective on lean thinking in the software development world. This is a must-read for leaders in all functional areas.”

*Arthur V. Hill, Lindahl Professor and Professor of Operations and Management Science,
Carlson School of Management, University of Minnesota*

“At OMG we have always believed that integration, repeatable processes and methodology, and high-quality, widely available standards were the missing links in the software world. Given the huge number of lessons to learn from other engineering and management sciences, it’s natural to apply the lessons of *Lean Manufacturing* to software production. John Schmidt has recognized the challenges and fought to integrate hard-won knowledge from other disciplines, and this book is a great example of what solid, clear, everyday lessons we can learn to make our organizations agile and innovative. Bravo!”

Richard Mark Soley, Chairman and CEO, Object Management Group, Inc.

“*Lean Integration* is invaluable to any business that relies on technological integration with its customers to expand. This book utilizes lean manufacturing principles to create successful software development projects in a replicable and measurable approach. By successful projects, I mean high quality, quick to production, maintainable for the long term, and under budget for both implementation and ongoing support. As an executive and a Six Sigma Black Belt of an expanding business process outsourcing company that relies on the integration of disparate customer systems for its growth and success, I believe the lean approach outlined in this book is the roadmap to follow.”

Howard L. Latham, Executive Vice President, API Outsourcing, Inc.

“*Lean Integration* is an excellent resource for anyone struggling with the challenges of performing integration for a complex enterprise. The authors have combined their experience to provide a practical roadmap for applying lean principles to the integration problem. If you are looking for an approach to tackle the integration chaos that exists in your environment, this book should be at the top of your reading list.”

Steve J. Dennis, Integration Competency Center Director, Nike

“As costs of raw technology decline, superior practice will dominate IT value. Increasingly, it’s not enough to be clever: it’s essential to be efficient, and that’s what John Schmidt and David Lyle will help IT practitioners do with their new book, *Lean Integration*. Point-to-point connections grow with (roughly) the square of the number of connected things, but Schmidt and Lyle offer a better way. Rising above the spaghetti bowl to treat integration as a scalable process, they make it practical for enterprise IT to make the most of complementary services in the cloud—promising the attentive reader huge improvements in IT economics.”

Peter Coffee, Director of Platform Research, Salesforce.com, Inc.

“*Lean Integration* is a practical discovery not an invention. For this reason everyone will eventually be doing it.”

Erwin Dral, Principal Product Manager, Informatica

“John Schmidt and David Lyle’s new book, *Lean Integration: An Integration Factory Approach to Business Agility*, is bound to shake up the software development industry. The authors show how to reduce costs and risks of software by applying lean management principles that force developers to focus on real customer/knowledge worker requirements to design quality into software the first time, from requirements definition to implementation and production operations. This is required reading for all information systems personnel who want to be on the cutting edge of quality management applied to software and systems engineering.”

Larry P. English, author, Information Quality Applied: Best Practices for Business Information, Processes and Systems

Lean Integration

*An Integration Factory Approach
to Business Agility*

John G. Schmidt

David Lyle

◆◆ Addison-Wesley

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States, please contact:

International Sales
international@pearsoned.com

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data

Schmidt, John G.

Lean integration : an integration factory approach to business agility
/ John G. Schmidt, David Lyle.

p. cm.

Includes index.

ISBN 978-0-321-71231-8 (pbk. : alk. paper)

1. Factory management. 2. Production management. 3. Business logistics. I. Lyle, David, 1964- II. Title.

TS155.S3234 2010

658.5—dc22

2010007196

Copyright © 2010 Informatica Corporation

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, write to:

Pearson Education, Inc.
Rights and Contracts Department
501 Boylston Street, Suite 9090
Boston, MA 02116
Fax: (617) 671-3447

ISBN-13: 978-0-321-71231-8

ISBN-10: 0-321-71231-5

Text printed in the United States on recycled paper at Courier in Stoughton, Massachusetts.

First printing, May 2010

To our loving and understanding families.

Contents

| | |
|-----------------------------------|--------------|
| <i>List of Figures and Tables</i> | <i>xv</i> |
| <i>Foreword</i> | <i>xix</i> |
| <i>Preface</i> | <i>xxiii</i> |
| <i>Acknowledgments</i> | <i>xxvii</i> |
| <i>About the Authors</i> | <i>xxix</i> |
| <i>Introduction</i> | <i>xxxi</i> |

Part I Executive Summary 1

Chapter 1 What Is Lean Integration and Why Is It Important? 3

| | |
|--|----|
| Constant Rapid Change and Organizational Agility | 5 |
| The Case for Lean Integration | 9 |
| What Is Integration? | 11 |
| Integration Maturity Levels | 14 |
| Economies of Scale (the Integration Market) | 16 |
| Getting Started: Incremental Implementation without “Boiling the Ocean” | 20 |

Chapter 2 A Brief History of Lean 23

| | |
|---|----|
| The Lean System | 29 |
| The Lean Practices | 34 |
| Lean Application Trends | 41 |
| Case Study: The Value of Lean in Service Industries | 44 |

Chapter 3 The Integration Factory 45

| | |
|---|----|
| What Is an Integration Factory? | 46 |
| The Integration Factory as an Element of an ICC | 52 |

| | |
|--|----|
| How Does the Integration Factory Work? | 55 |
| Integration Factories as Self-Service ICCs | 64 |

Part II Applying Lean Principles 67

Chapter 4 Focus on the Customer and Eliminate Waste 69

| | |
|--|----|
| Focus on the Customer | 70 |
| Integration Wastes | 74 |
| Case Study: Waste Elimination at Clicks-and-Bricks | 81 |
| Case Study: Waste Elimination at Big Bank | 85 |
| Focus on the Integration Value Chain | 87 |

Chapter 5 Continuously Improve 89

| | |
|---|----|
| Continuous Learning and Knowledge Management | 90 |
| Case Study: Continuous Improvement at Clicks-and-Bricks | 91 |

Chapter 6 Empower the Team 103

| | |
|---|-----|
| What Is a Team? | 104 |
| Examples of Empowered Teams in Software | 107 |
| Creating an Empowered Lean Integration Team | 109 |
| Leadership and Vision | 112 |
| Important Practices That Help Enable Empowered Teams | 117 |
| Organizing the Team: Thoughts on Organizational Structures | 120 |
| Case Study: Smith & Nephew—Integrating Lean Principles with Data Quality | 122 |

Chapter 7 Optimize the Whole 131

| | |
|--|-----|
| Optimize the Whole Rather than Optimize the Parts | 132 |
| What Is “the Whole”? An Introduction to Value Stream Mapping | 134 |
| Selecting Metrics to Optimize the Whole | 139 |

Chapter 8 Plan for Change and Mass-Customize 145

| | |
|---|-----|
| Techniques for Enabling Constant Change | 146 |
| Mass Customization | 152 |
| Case Study: Using Mass Customization | 159 |

| | | |
|-------------------|---|------------|
| Chapter 9 | Automate Processes and Deliver Fast | 163 |
| | Pitfalls of Automation—Building “Stuff” Faster | 164 |
| | Delivering Fast | 167 |
| | Automating Processes—Using the Computer to Make Complexity Manageable | 169 |
| | Case Study: Automating Processes at Wells Fargo Home Mortgage | 174 |
| Chapter 10 | Build Quality In | 181 |
| | Two Areas of Quality: Data Quality and Integration Quality | 182 |
| | Quality Evolution and Lean | 184 |
| | Data Quality | 185 |
| | Integration Quality | 192 |
| | Case Study: Building Quality In at a Utility Company ICC | 198 |
| Part III | Implementation Practices | 203 |
| Chapter 11 | Financial Management | 205 |
| | Challenges | 207 |
| | Activities | 214 |
| | Business Case Development | 215 |
| | Case Study: A “Creating the Wave” Investment Strategy | 236 |
| | Case Study: Enterprise Data Warehouse Rationalization | |
| | Business Case | 238 |
| | Chargeback Accounting | 240 |
| | Chargeback Case Studies | 250 |
| Chapter 12 | Integration Methodology | 253 |
| | Activities | 256 |
| | Agile versus Lean Methodology | 263 |
| | Case Study in Simplicity: The Architecture of the Web and REST versus SOA | 269 |
| | Engagement Services Management | 271 |
| | Case Study: Integration Methodology in a Decentralized Enterprise | 274 |

| | | |
|-------------------|---|------------|
| Chapter 13 | Metadata Management | 281 |
| | Metadata Scope for Lean Integration | 284 |
| | Metadata Management Framework | 285 |
| | Challenges | 289 |
| | Prerequisites | 292 |
| | Industry Practices | 293 |
| | Activities | 295 |
| | | |
| Chapter 14 | Information Architecture | 301 |
| | Challenges | 304 |
| | Prerequisites | 308 |
| | Activities | 309 |
| | Methodology | 310 |
| | Information Architecture Models | 312 |
| | Data at Rest | 317 |
| | | |
| Chapter 15 | Business Process Management | 321 |
| | Data-in-Motion Models | 324 |
| | Activities | 326 |
| | Architecture | 328 |
| | Case Study: The Post Closing Systems Architecture | 330 |
| | | |
| Chapter 16 | Modeling Management | 333 |
| | Service-Oriented Architecture Can Create a New Hairball | 336 |
| | Challenges | 339 |
| | Coupling and Cohesion Framework | 343 |
| | Canonical Modeling Techniques | 345 |
| | Navigating the Modeling Layers | 350 |
| | Activities | 352 |
| | Case Study: European Interoperability Framework | 357 |
| | Case Study: Object-Relational Mismatch | 359 |
| | | |
| Chapter 17 | Integration Systems | 361 |
| | What Is an Integration System? | 362 |
| | Integration Systems Taxonomy | 364 |
| | Challenges | 369 |

| | |
|--|------------|
| Industry Practices | 370 |
| Activities | 371 |
| Portfolio Rationalization | 378 |
| Appendix A Lean Terminology | 387 |
| Appendix B Integration Laws | 395 |
| Law #1: The Whole Is Greater than the Sum of Its Parts | 395 |
| Law #2: There Is No End State | 396 |
| Law #3: There Are No Universal Standards | 396 |
| Law #4: Information Adapts to Meet Local Needs | 397 |
| Law #5: All Details Are Relevant | 397 |
| Appendix C Glossary | 399 |
| Common Acronyms | 399 |
| Definitions | 400 |
| Index | 409 |

Figures and Tables

| | | |
|-------------------|---|-----|
| <i>Figure 1.1</i> | <i>Definition of integration</i> | 13 |
| <i>Figure 1.2</i> | <i>Evolution of integration technology and management practices</i> | 15 |
| <i>Figure 1.3</i> | <i>Diminishing returns and economies of scale</i> | 19 |
| <i>Figure 2.1</i> | <i>Manufacturing management system chronology</i> | 25 |
| <i>Figure 2.2</i> | <i>Toyota “House” of Lean</i> | 31 |
| <i>Figure 2.3</i> | <i>Lean Enterprise Management System</i> | 32 |
| <i>Figure 2.4</i> | <i>Value stream mapping</i> | 37 |
| <i>Figure 2.5</i> | <i>Lean Enterprise Management System problem-solving process</i> | 38 |
| <i>Figure 2.6</i> | <i>Applying Lean to nonmanufacturing business functions</i> | 41 |
| <i>Figure 3.1</i> | <i>Four flow types in an Integration Factory</i> | 49 |
| <i>Figure 3.2</i> | <i>Example factory process</i> | 56 |
| <i>Figure 3.3</i> | <i>Integration team interactions</i> | 58 |
| <i>Figure 3.4</i> | <i>Development factory information flow tools and repository</i> | 63 |
| <i>Figure 3.5</i> | <i>ICC organizational models</i> | 64 |
| <i>Table 5.1</i> | <i>Integration Development Cost Projections Based on Custom Project Silo Approach</i> | 95 |
| <i>Table 5.2</i> | <i>Integration Development Cost Projections Based on Integration Factory</i> | 95 |
| <i>Table 5.3</i> | <i>Business Case Results after Three Years of Integration Factory Operation</i> | 100 |
| <i>Figure 6.1</i> | <i>Model for team development and performance</i> | 107 |
| <i>Figure 6.2</i> | <i>ICC organizational structure</i> | 121 |
| <i>Figure 6.3</i> | <i>Data quality dimensions</i> | 126 |

| | | |
|-------------|---|-----|
| Figure 7.1 | Value stream map for an integration change request | 137 |
| Figure 7.2 | An alternative value stream map for an integration change request | 138 |
| Figure 9.1 | Integration problem tickets closed by month | 172 |
| Table 9.1 | Before and After Metrics for HUD Review of FHA-Insured Loans | 177 |
| Figure 10.1 | Good Energy enterprise subject area model | 200 |
| Table 11.1 | Financial Management Maturity Levels | 206 |
| Figure 11.1 | Demand and supply mismatch | 210 |
| Figure 11.2 | Business case development steps | 217 |
| Figure 11.3 | Sample factory business case graph | 227 |
| Table 11.2 | Industry Cost Benefits Metrics | 228 |
| Figure 11.4 | Sample MBF charts | 239 |
| Table 11.3 | MBF Action Plan for EDW Rationalization Project | 240 |
| Figure 11.5 | Economic framework | 242 |
| Figure 11.6 | Chargeback models | 244 |
| Table 11.4 | Project-Based Direct Purchase | 245 |
| Table 11.5 | Tiered Flat Rate Chargeback for Enterprise License Scenario | 245 |
| Table 12.1 | Integration Methodology Maturity Model | 255 |
| Figure 12.1 | Integration methodology activities | 256 |
| Table 12.2 | Sample 120-Day ICC Implementation Plan | 259 |
| Table 12.3 | Agile Principles Compared to Lean Principles | 264 |
| Figure 12.2 | Procure-to-pay process | 277 |
| Table 13.1 | Metadata Management Maturity Model | 283 |
| Figure 13.1 | Metadata dimensions | 284 |
| Figure 13.2 | Metadata context diagram | 285 |
| Figure 13.3 | Information architecture framework | 287 |
| Table 13.2 | Objection Handling with Metadata Management | 291 |
| Figure 13.4 | Metadata management implementation steps | 295 |
| Table 14.1 | Information Architecture Maturity Model | 302 |
| Figure 14.1 | Information architecture within a data governance methodology | 311 |

| | | |
|--------------------|--|-----|
| <i>Figure 14.2</i> | <i>Information architecture models</i> | 313 |
| <i>Figure 14.3</i> | <i>Data-at-rest models</i> | 317 |
| <i>Figure 14.4</i> | <i>Information architecture team interactions</i> | 320 |
| | | |
| <i>Table 15.1</i> | <i>Business Process Management Maturity</i> | 322 |
| <i>Figure 15.1</i> | <i>Data-in-motion models</i> | 325 |
| <i>Figure 15.2</i> | <i>BPM architecture for channel integration or straight-through processing</i> | 329 |
| | | |
| <i>Table 16.1</i> | <i>Modeling Management Maturity Model</i> | 335 |
| <i>Figure 16.1</i> | <i>Services in an SOA creating a new integration hairball</i> | 337 |
| <i>Figure 16.2</i> | <i>Canonical models and data services in an SOA</i> | 338 |
| <i>Table 16.2</i> | <i>Loose Coupling Practices and Their Cost Implications</i> | 341 |
| <i>Figure 16.3</i> | <i>Coupling and cohesion relationship</i> | 344 |
| <i>Figure 16.4</i> | <i>Canonical techniques in relation to the coupling-cohesion framework</i> | 347 |
| <i>Figure 16.5</i> | <i>Modeling management required for data services support</i> | 351 |
| <i>Figure 16.6</i> | <i>Modeling management 11-step process</i> | 354 |
| <i>Figure 16.7</i> | <i>Graphical representation of the European Interoperability Framework</i> | 358 |
| <i>Figure 16.8</i> | <i>Semantic example of object-relational impedance mismatch</i> | 360 |
| | | |
| <i>Figure 17.1</i> | <i>Conceptual view of information exchange between business systems</i> | 363 |
| <i>Figure 17.2</i> | <i>Example of a “simple” integration system</i> | 364 |
| <i>Figure 17.3</i> | <i>Example of a complex integration system including a variety of components</i> | 365 |
| <i>Figure 17.4</i> | <i>Taxonomy of integration systems</i> | 366 |
| <i>Table 17.1</i> | <i>Integration Systems Maturity Model</i> | 368 |
| <i>Figure 17.5</i> | <i>Steps to establish a sustaining integration systems capability</i> | 371 |
| <i>Figure 17.6</i> | <i>ICC functional scope framework</i> | 374 |
| <i>Figure 17.7</i> | <i>Example ICC functional scope</i> | 376 |
| <i>Figure 17.8</i> | <i>Seven-step portfolio rationalization methodology</i> | 380 |
| <i>Table 17.2</i> | <i>Application Rationalization Table</i> | 382 |
| <i>Figure 17.9</i> | <i>Infrastructure migration road map</i> | 383 |
| | | |
| <i>Figure A.1</i> | <i>MBF template</i> | 390 |

Foreword

More than ten years ago I wrote what many call the “definitive book on integration” entitled *Enterprise Application Integration*. The idea for the book was simple, really. Put some time and energy around planning how various enterprise systems communicate with each other, and leverage some sophisticated technology to make integration work in reliable and changeable ways.

Until then, and what is still sometimes the case today, many looked upon integration as a one-off development project, coding to interfaces between two or more systems. It was always cheaper, it always worked at first, but it always hit a brick wall at some point. This was a hack-after-hack approach that quickly led to a dysfunctional state of architecture where changes are difficult if not impossible to make. We needed a better approach and some good thinking around how integration is done.

While I often get credit for kicking off integration as an architectural discipline, the reality is that many smart people worked on the integration problem prior to my book, and they are still working on it today—for instance, the authors of this book, John and David. I remember meeting John for breakfast in 1998 when he was working for AMS and talking about what was next for integration. Even then John’s thinking was highly innovative and forward-looking. John’s most profound ideas placed discipline around integration, something that was not the case then and is still lacking today.

So, what is needed today? First and foremost is the importance of data in this generation of cloud computing and SOA. Data has always been, and always will be, the foundation of all sound architecture, no matter if you leverage SOA as an approach to architecture, or cloud computing as an option for platform deployment. Understanding the importance of data means that you’ll create more efficient and agile architectures that can accommodate any required changes to the business.

In the last ten years, we've gone from hand-coding interfaces between source and target systems to EAI as a better approach to integration. Now we move on to SOA as a way to create architectures that address most behaviors and information as sets of services, and to data-oriented SOAs that are "Lean Integration and Integration Factories." In many respects we are returning to our roots, but doing so with better technology and refined approaches, such as Lean Integration. The benefits will be a reduction in system costs and a huge increase in efficiencies.

The fact of the matter is that integration is an architectural pattern. Like any architectural pattern, you can improve and refine integration into something more productive and more innovative. That is exactly what the authors have done here. In short, John and David have written the right book, at the right time, for the right reasons. John and David present concepts that take integration to the next level, making integration more accessible, efficient, and cost-effective. I jumped at the chance to promote this book to my publisher, as well as the chance to write the foreword. *Lean Integration* should be read by anyone involved with an integration project.

"Lean Integration" is a management system that emphasizes continuous improvements, meaning you don't complete the links and call it done. Integration requires an ongoing interest in the way integration is carried out and the mechanisms required. This means consistently reevaluating and improving the approaches we leverage for integration, as well as the technology employed. Integration is a journey, not a project.

The end results of leveraging Lean Integration go right to the bottom line, including as much as a 50 percent labor productivity improvement via value stream mapping. This is the use of constant improvement to locate and eliminate non-value-added activities, in short, things that are not needed and don't add value to the ultimate customer.

Moreover, Lean Integration provides the value of agility. This means you create an integration infrastructure that is able to change around a changing business or mission. You're able to quickly adjust around these changes, and thus you have the value of quickly altering the business to get into new markets, get out of old markets, and outmaneuver your competition. The strategic advantages of agility are huge.

The approach of addressing data quality within Lean Integration means that you can finally treat data as what it is: an asset. Not addressing data quality means not taking care of that asset, and thus the diminished value of the

data results in the diminished value of the business. Data quality also addresses the use of data governance, which is required to adhere to regulations but needed more to protect our data assets no matter what systems are currently managing the data.

Core to this concept is the ability to promote and manage innovation, allowing those in the organization to create and test their own ideas for improving integration. This provides a feeling of empowerment, which is a benefit to the employee, as well as the actions around good ideas, which is a benefit to the company.

What is most profound about *Lean Integration* is that many of the ideas in this book are obvious, but not leveraged. In reading this book you find many things that seem to be simple but good ideas, and you find yourself asking, “Why did I not think of that?” over and over again. However, while the ideas are important, having a framework of understanding is vital as well. You have to place these disciplines into context and make them part of a repeatable process, which is another core feature of this book.

Lean Integration marks the end of chaotic integration practices and the beginning of continuous improvement, which enhances the value of integration. As the number of systems increases within enterprises, so does the need for integration. You can do integration the right way, which is the Lean way. Or you can struggle with it for years to come. I recommend the former.

David S. Linthicum

Preface

By John Schmidt

I have been practicing Lean Integration for over 15 years. I just didn't know it.

Over the past 30 years I've learned various Lean practices such as *kaizen* (continuous improvement) and closely related methods such as Six Sigma and agile software development. But I didn't fully understand Lean as a system until I studied it more thoroughly in 2008. The surprising discovery was that I had in fact been applying Lean principles for many years without formally identifying them as such.

The reality is that Lean principles such as customer intimacy, continuous improvement, seeking quality perfection, and developing effective teams are common sense. David and I don't have a monopoly on common sense, so we expect that as you read this book you will see examples of Lean principles in your own life. We hope that this book does at least two things for you: first, that it enriches your own experiences with more examples, case studies, and practical advice so that you can improve your level of competence; and second, that it provides a more complete management "system" that makes all these commonsense ideas teachable so that you can improve the level of competence of your team and throughout your value chain.

Contrary to common practices I have always viewed integration as a repeatable ongoing process rather than as a custom one-off project effort. While an integration point between any two systems always looks unique from the perspective of the two systems, if you step back and look at all the information exchanges between systems in an enterprise, what you find is a relatively small number of patterns that are repeated over and over again. The details are different for every instance (at a minimum the data itself that is being exchanged is unique) but the patterns are not.

I have been fortunate over the years in my work at the Integration Consortium, Wells Fargo Bank, Bank of America, Best Buy, American Management Systems, and Digital Equipment Corporation to have the opportunity to help hundreds of companies implement integration practices. The first successful Integration Factory I implemented (several prior attempts didn't quite take off) was an exercise in mass customization—building customized message adapters on top of vendor platforms with reusable components based on a service delivery model that focused on customer needs. The next successful Integration Factory encompassed not only real-time application-to-application integration, but also high-volume batch-oriented database-to-database integration, external business-to-business (B2B) managed file transfer, and business process integration. The factory also included an internal Web application that allowed users (customers of the integration team) to interact with the factory through a series of role-based user interfaces.

Through these and hundreds of other experiences over the years I have developed a perspective on what it takes to implement sustainable integration practices. The first book on which David and I collaborated, *Integration Competency Center: An Implementation Methodology*,¹ articulated the concepts that make the difference between a successful and an unsuccessful integration strategy. This book takes Integration Competency Centers (ICCs) to the next level by adding more specific best practices and a rich collection of case studies, and by leveraging the vast body of knowledge that has developed over the past 50 years on Lean practices. The net result is sustainable integration that begins to turn an art into a science by making it teachable and repeatable.

By David Lyle

One of the most rewarding professional periods of my life was leading a remarkable team of developers who put together packaged analytic applications to sell as software products about ten years ago. Little did I realize that our team, at the high-water mark numbering over 80 people, was employing all of the Lean Integration principles discussed in this book while

1. John G. Schmidt and David Lyle, *Integration Competency Center: An Implementation Methodology* (Informatica Corporation, 2005).

putting together what John and I now describe as an Integration Factory. Like John, I didn't realize we were employing Lean thinking; I just thought we were continually benefiting from people's innovative ideas to work smarter, not harder.

Our development team made the realization that the integration logic in those analytic applications followed a relatively small number of integration or processing patterns. Because the time spent on the design, implementation, and testing of integration was such a large proportion of the total development costs, the team realized we became significantly more effective if we thought in terms of developing assembly lines around these integration patterns rather than crafting all integration logic as "unique works of art." By changing our entire approach over the course of four years, we became far more efficient as a development organization, but most important, we developed higher-quality, more maintainable products for our customers.

Over the past several years, John and I have worked at or talked with numerous companies around the world about how to develop and grow their ICCs. We found several ICCs achieving great success by automating certain processes, using mass customization techniques, or adopting agile development approaches. Several we call out explicitly as case studies in this book, but many of the ideas in this book are the products of conversations and achievements of numerous integration professionals we've spoken with over the years. In other words, besides our own experiences, John and I have seen others be successful with many of the ideas we're pulling together in this book.

That being said, we don't mean to ever imply that adopting these ideas is easy. Integration is an especially complex, challenging problem, both technically and organizationally. All companies have had to spend significant time continually convincing executive management of the benefits of attacking integration as a discipline that is part of the overall enterprise architecture, rather than as a temporary exercise that is unique to each project. Most IT executives are less aware of the detailed costs of integration or benefits of ICCs.

Lean Integration and the Integration Factory are neither destinations nor vendor products; implementing them is a journey that takes many years. Despite the fact that John and I both now work for Informatica, we worked hard to make this book vendor-neutral. We wanted the book to be broadly useful to integration professionals rather than to be based on a

specific vendor's software offering. Successful ICCs are a product of the synergy of good people, effective processes, and appropriate technology. This book represents what we've learned over the years from so many people about how Lean thinking can make ICCs significantly more efficient and effective for their customers.

Acknowledgments

This book is itself an example of Lean practices. For starters, it was produced by a *value stream*. Over the years there have been countless individuals in many roles and functions who have contributed what we hope meets the expectations of our customers—you. The book demonstrates *pull* in the sense that we wouldn't have written it if we hadn't received numerous requests from readers of our first book asking for more detailed and prescriptive techniques. It demonstrates *just-in-time* in that Lean thinking is on a rapid growth curve for adoption across many nonmanufacturing business processes; two years ago the book would have been premature for broad-based adoption, and two years from now there will likely be many books on how to apply Lean to IT processes. It also demonstrates *continuous improvement*; many ideas in this book were first written as blog articles that resulted in reader feedback, which has improved the quality of the final result.

To fully acknowledge all the individuals and organizations that have contributed to the culmination of Lean Integration could double the length of this book but would not add value to our readers. So we apologize in advance for not mentioning all of the support and contributions we have received over the years. We would, however, like to mention a few noteworthy contributions that we hope will add some value.

The detailed practical examples from real-world situations add a level of depth to the book, and value to our readers, for which we are profoundly grateful. Many thanks go out to the following people for their case study contributions:

- Todd Soller for his support as well as the many people who accomplished the impressive results: Lucas Anderson, Anil Atri, Clark Becker, Charles Betz, Michael Frank, Greg Friesen, Arul James Suneela Kanuri, Mark

Karger, Todd Lauinger, Radhika Menon, Lelanie Moll, Balaji Ramaswamy, Rick Sorenson, Bob Steel, and Raghu Venkateshwar

- Michael Levine for the Wells Fargo Post-Closing case study, an amazing example of cycle-time reduction and quality improvement through effective flow-through processes
- Barbara Latullipe for her leadership and insights in applying Lean practices to data quality and master data management at Smith & Nephew
- Gary Kamath and Manish Varma for providing valuable contributions to the case studies
- Patrick Kirkes for his skills in mass customizing integration logic
- Steve Morgan for his persistence in implementing domain models; in conjunction with his organization's integration hub, the approach provided an impressive ability to mass-customize data in support of organizational agility

Many thanks, in no particular order, also go out to

- Dr. Art Hill, professor at the Carlson School of Management at the University of Minnesota, for his lessons in Lean and for giving us the idea for the title
- Paul Husby, author of Chapter 2 and most of the terminology in Appendix A, for helping us paint a rich picture of the Lean system as well as providing a history of Lean
- Sohaib Abbasi, CEO at Informatica, for constantly nagging us to write a sequel to the original ICC book
- Brian Hodges, Chris Boorman, Paul Hoffman, and James Markarian, senior executives at Informatica, for their strong support and for giving us the company time, resources, and creative freedom to develop a compelling message
- Kevin Davis, Stephen Dulzer, Jill Dyche, and Tom Poppendieck, for their detailed reviews and comments on the draft manuscript; the book is 100 percent better because of their constructive feedback
- David Linthicum, for encouraging us to write the book and introducing us to the publisher
- Tiffany Gumfory, Jeremy Phelps, and Alison Spong, for handling many of the logistical details associated with simply "getting it done"
- Michael Kuhbock and Leanne MacDonald, for their feedback and providing the support of the Integration Consortium to help make it happen

About the Authors



John Schmidt's integration career began over 30 years ago when he was a hardware technician at Digital Equipment Corporation. He tells the story about plugging a PDP-11 cable into the wrong socket, which put 30 volts rather than 5 volts on the bus and fried every chip connected to it. Hence the first lesson in integration: While the plug might fit in the socket, hidden incompatibilities in behavior that are not readily visible can be disastrous.

John went on to work as a software engineer, project manager, sales representative, professional services manager, program manager, and enterprise architect. He has practiced and honed his integration expertise in half a dozen industries (banking, retail, telecommunications, education, government, and utilities) and in just as many countries. John was director and chairman of the Integration Consortium for eight years, has written numerous articles on systems integration and enterprise architecture, developed program management practices, and is a frequent speaker at industry conferences.

John's current role is vice president of Global Integration Services at Informatica. He advises clients on the business potential of emerging technologies; leads in the creation of strategies for enterprise initiatives; and plans, directs, and supervises Informatica's Integration Competency Center Practice. He graduated from Red River College and holds a master's degree in business administration from the Carlson School of Management at the University of Minnesota.



David Lyle's career has traveled from the hardware world of computer design to the software world of product development, and along the way he developed early massively parallel (MPP) UNIX systems and associated parallel RDBMS systems for Unisys. While working on the field implementations of these large-scale data warehousing systems, he began noticing consistent, recurring patterns.

From these experiences, David helped found Influence Software, one of the early packaged analytic applications companies. Informatica bought Influence Software at the end of 1999, and for three years David led the R&D organization that built the packaged business intelligence content, the data models, and the preconfigured data integration logic for extracting data out of the most common ERP and CRM systems. After Informatica divested itself of these packaged applications to focus on its core business of data integration, David turned his attention to guiding Informatica's product direction, based upon the "after-market" tools and utilities his earlier organization had used to automate common integration patterns or solve many of the challenges they faced. These ideas culminated in Informatica's Metadata Manager, automated generation of integration logic following configurable architecture patterns, and patents on using canonical data virtualization techniques within integration architectures to move toward elimination of point-to-point integration approaches.

David is currently vice president of Product Strategy, where he continues to guide the longer-term vision for how the Informatica platform assists organizations in integrating and managing their data more efficiently and effectively. He graduated from Stanford University with a bachelor's of science in electrical engineering.

Introduction

Knowledge is power.

Sir Francis Bacon¹

Lean Integration is a management system that emphasizes continuous improvement and the elimination of waste in end-to-end data integration and process integration activities. Lean practices are well established in other disciplines such as manufacturing, supply chain management, and software development to name just a few, but the application of Lean to the integration discipline is new.

Lean manufacturing is a management system that emphasizes creating value for end customers and eliminating activities that are not value-added (waste). Its principles were derived from the Toyota Production System (TPS), which was developed over 50 years ago but since the 1990s has simply been referred to as Lean. While Lean is rooted in product manufacturing, it is now widely regarded as a management approach that can be applied effectively to a wide range of product and service industries. Lean is closely related to, and borrows from, other methodologies, including Value Network, Theory of Constraints, Six Sigma, and Statistical Process Control (including the work of W. Edwards Deming).

Lean software development is an agile approach that translates Lean manufacturing principles and practices for the software development domain. It was adapted from the TPS and introduced by Mary and Tom Poppendieck in their book *Lean Software Development* and expanded in

1. Sir Francis Bacon, *Religious Meditations of Heresies*, 1597.

Implementing Lean Software Development, followed by *Leading Lean Software Development* in 2009.²

Lean Integration builds on these prior works by applying their principles to the process of integration. The definition of *integration* used in this book is “the practice of making independent applications work together as a cohesive system on an ongoing basis.” While there are myriad integration technologies and information exchange patterns, broadly speaking, integration solutions typically fall into one of two styles:

1. **Process integration:** automation of processes that cut across functional or application boundaries where process state needs to be maintained independently of the underlying application systems or where multiple data consumers or data providers need to be orchestrated as part of a business transaction
2. **Data integration:** accessing data and functions from disparate systems to create a combined and consistent view of core information for use across the organization to improve business decisions and operations

Another integration style that some practitioners call out as a separate category is service integration as part of a service-oriented architecture (SOA), where application functions are separated into distinct units, or services, that are directly accessible over a network in a loosely coupled manner and may be orchestrated with stateless interactions. The contrary argument is that service integration and the associated infrastructure of an enterprise service bus (ESB) are simply characteristics of a given process integration or data integration design.

Regardless of how many integration categories there are, Lean Integration applies to all of them. Software systems are, by their very nature, flexible and will change over time (legacy systems are often an exception). Interfaces and information exchanges between systems are never built just once. Integration therefore is not a one-time activity; it is ongoing. In summary, therefore, Lean Integration is the application of Lean principles and methods to the challenges of process and data integration on a sustainable basis.

2. Mary and Tom Poppendieck, *Lean Software Development: An Agile Toolkit* (Addison-Wesley, 2003); *Implementing Lean Software Development: From Concept to Cash* (Addison-Wesley, 2007); *Leading Lean Software Development: Results Are Not the Point* (Addison-Wesley, 2009).

This book builds on the first book that we wrote in 2005, *Integration Competency Center: An Implementation Methodology*.³ Integration Competency Centers (ICCs) continue to be a key ingredient of efficient and sustainable integration, and the core concepts in the 2005 book remain applicable to a Lean Integration practice. It is not necessary to read the ICC book first, but it still serves as an effective primer. This book adds depth to ICC principles and methods and extends the concepts to a broader view of the value chain.

One of the things we have learned since writing the first book is that an ICC by any other name is still an ICC. For example, some groups have names such as

- Integration Solutions Group (ISG)
- Center of Excellence or Center of Expertise (COE)
- Business Intelligence Competency Center (BICC)
- Data Quality COE
- Enterprise Data Warehouse (EDW)
- SOA COE
- Center of Competency (CoC)

And the list goes on. The point is that the principles and methods presented here are management practices that apply to all of them. Of course, the subject area expertise is different for each one, as are the scope, the technical skills needed by the staff, the specific tools and technologies that are used, and the internal and external customers that are served by it. In any event, we need to have a name for the organizational unit and need to call it something in this book, so we will continue to use Integration Competency Center, or ICC for short, as the umbrella term for all varieties. For the record, our definition of *Integration Competency Center* is as follows:

A permanent cross-functional team operating as a shared-services function supporting multiple organizational units and sustaining integration solutions in a coordinated manner

This book is divided into three parts. Part I serves as a summary for executives of Lean Integration and therefore provides an overview for a broad

3. John G. Schmidt and David Lyle, *Integration Competency Center: An Implementation Methodology* (Informatica Corporation, 2005).

audience from senior IT executives to front-line operations staff. It provides an overview of and justification for Lean by answering questions like these:

- “Why Lean?” and “So what?”
- “As a business executive, what problems will it help me solve?”
- “As an IT leader or line-of-business owner, why am I going to make a considerable investment in Lean Integration?”
- “How is this different from other methods, approaches, and frameworks?”
- “Why am I as an IT professional going to embrace and sell Lean Integration internally?”

This first part also provides an overview of Lean practices, where they come from, and how they have evolved. It includes insightful research and current trends in how Lean is being adapted to different industries and management disciplines.

Part I concludes with an overview of the Integration Factory—the next-generation integration technology that adds a high degree of automation to the flow of materials and information in the process of building and sustaining integration points. Examples of automation include requirements definition, code generation, testing, and migration of code objects from development to test to production environments. The Integration Factory, we believe, will be the dominant new “wave” of middleware for the next decade (2010s). It views the thousands of information exchanges between applications in an enterprise as mass customizations of a relatively small number of patterns.

The management practice that optimizes the benefits of the Integration Factory is Lean Integration—the use of *Lean principles* and *tools* in the process of making independent applications work together as a cohesive system. The combination of factory technologies and Lean practices results in significant and sustainable *business benefits*.

Part II introduces the seven Lean Integration principles that optimize the Integration Factory and shows how they can be applied to the challenges of system, data, and application integration in a sustainable fashion. This section of the book is targeted at business and IT leaders who are implementing, or considering implementing, a Lean Integration program. Each chapter in this part focuses on one of the seven core principles:

1. **Focus on the customer and eliminate waste:** Maintain a spotlight on customer value and use customer input as the primary driver for the

development of services and integration solutions. Waste elimination is related to this principle, since waste consists of activities that don't add value from the customer's perspective rather than from the supplier's perspective. Related concepts include optimizing the entire value stream in the interest of things that customers care about and just-in-time delivery to meet customer demands.

2. **Continuously improve:** Use a data-driven cycle of hypothesis-validation-implementation to drive innovation and continuously improve the end-to-end process. Related concepts include how to amplify learning, institutionalizing lessons learned, and sustaining integration knowledge.
3. **Empower the team:** Share commitments across individuals and multi-functional teams and provide the support they need to innovate and try new ideas without fear of failure. Empowered teams and individuals have a clear picture of their role in the value chain, know exactly who their customers and suppliers are, and have the information necessary to make day-by-day and even minute-by-minute adjustments.
4. **Optimize the whole:** Make trade-offs on individual steps or activities in the interest of maximizing customer value and bottom-line results for the enterprise. Optimizing the whole requires a big-picture perspective of the end-to-end process and how the customer and enterprise value can be maximized even if it requires sub-optimizing individual steps or activities.
5. **Plan for change:** Apply mass customization techniques to reduce the cost and time in both the build and run stages of the integration life cycle. The development stage is optimized by focusing on reusable and parameter-driven integration elements to rapidly build new solutions. The operations stage is optimized by leveraging automated tools and structured processes to efficiently monitor, control, tune, upgrade, and fix the operational integration systems.
6. **Automate processes:** Judiciously use investments to automate common manual tasks and provide an integrated service delivery experience to customers. In mature environments, this leads to the elimination of scale factors, the ability to respond to large integration projects as rapidly and cost-effectively as small changes, and the removal of integration dependencies from the critical implementation path.
7. **Build quality in:** Emphasize process excellence and building quality in rather than trying to inspect it in. Related concepts include error-proofing the process and reducing recovery time and costs.

Part III is intended for those who have direct responsibility for implementing an integration strategy or are members of an integration team and are interested in improving their ICC or Lean skills. It provides detailed best practices, grouped into seven competency areas. These competencies are ongoing capabilities that an organization needs in order to provide a sustainable approach to integration.

1. **Financial management:** Financial management is a vital competency of a Lean practice that is operating a shared-services group; it takes more than technical activities to create and sustain an effective shared-services group. By showing how to articulate the business value of technology, we are seeking to ensure that the team does not become isolated from the business environment that sustains it. Credibility is established not only by operating a successful competency team, but also by being perceived by business leaders as doing so.
2. **Integration methodology:** An integration methodology defines the life cycle of dependencies that are involved in building a Lean Integration team to the point that it becomes an ongoing governing body, able to sustain the integration achieved from specific projects. The integration methodology is concerned with not just building quality solutions, but more important with setting up the processes to sustain solutions indefinitely in a production environment.
3. **Metadata management:** The ability to manage metadata is essential for managing data as an asset, and it is an enabler for a broad range of process and data integration programs such as straight-through processing, master data management, and data governance. Metadata is conceptually simple but challenging to implement in large, complex organizations. Essentially it is documentation and business rules about data in terms of what it means; where it is located; how it is accessed, moved, and secured; who is responsible for it; and who is allowed to use it.
4. **Information architecture:** Information architecture is an element of a broader enterprise architecture capability. Architecture and integration are complementary enterprise practices that intersect most significantly in the information domain. Architecture is about differentiating the whole and transforming the business, whereas integration is about assembling the parts into a cohesive, holistic system. One discipline takes a top-down approach while the other is bottom-up; both are essential.

5. **Business process management:** Business process management (BPM) is a method of efficiently aligning an enterprise with the needs of its customers. It is a holistic management approach that promotes business effectiveness and efficiency while striving for innovation, flexibility, and integration with technology. Business processes are a prime integration point since this is often where disparate functions come together and interact to share data in the interests of maximizing customer satisfaction and enterprise effectiveness.
6. **Modeling management:** Modeling management is the discipline for defining, using, and maintaining a variety of views (simplified abstractions) of the enterprise and its data in support of integration strategies. One of the critical needs within all large corporations is to achieve efficient information exchanges in a heterogeneous environment. The typical enterprise has hundreds of applications that serve as systems of record for information that, although developed independently and based on incompatible data models, must share information efficiently and accurately in order to effectively support the business and create positive customer experiences.
7. **Integration systems:** This competency addresses the need for managing the life cycle of integration technology and systems as a distinct class of application that provides a sustainable operating infrastructure to support the free flow of information in an organization in an integrated manner.

PART I

Executive Summary

Alice: *Would you tell me, please, which way I ought to go from here?*

The Cat: *That depends a good deal on where you want to get to.*

Alice: *I don't much care where.*

The Cat: *Then it doesn't much matter which way you go.*

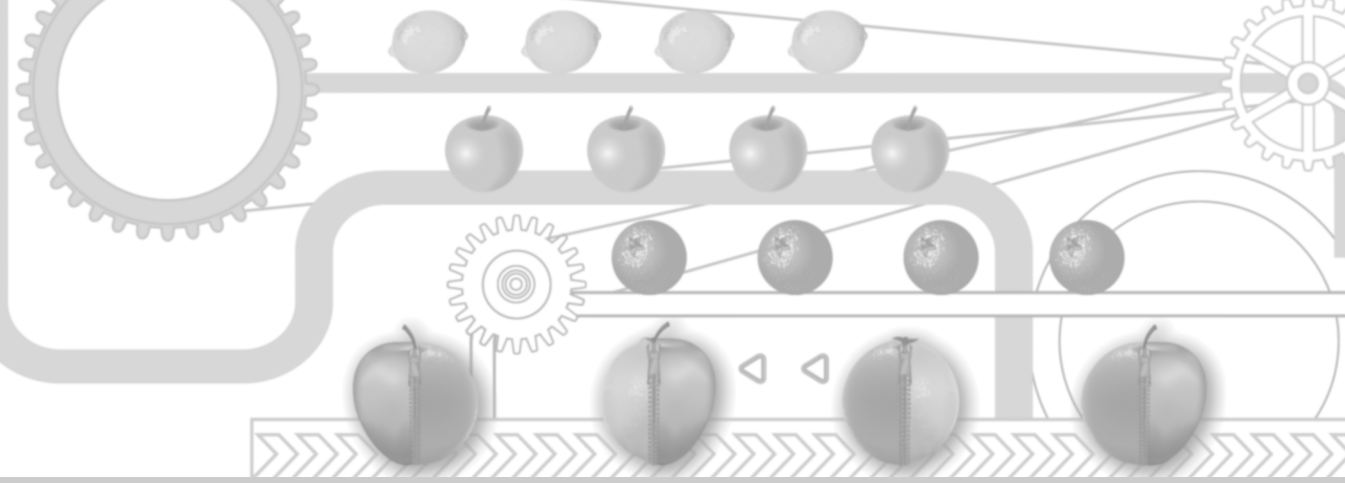
Lewis Carroll, *Alice's Adventures in Wonderland*

As the cat in Lewis Carroll's story so clearly advises, if you don't know your destination, it doesn't matter which path you take. This executive overview is intended to paint a picture of the destination for an enterprise integration strategy. Parts II and III of the book then illuminate the path.

The size of this book may be daunting to some readers, but Part I shouldn't be. We structured the book in three parts specifically to make the subject of Lean Integration more approachable for different audiences. Part I is designed to be read by anyone, not just executives. Front-line staff should read Part I to understand the big-picture context, and senior executives too should read it to appreciate the role they play in this new approach and to internalize the strategic framework.

This executive overview can be read as a stand-alone publication. It provides a solid overview of the business justification for Lean Integration, the history of Lean and how it has evolved to this point, and how its principles can be applied in the context of an Integration Factory. Readers who are primarily interested in understanding the concepts and strategic framework and will be relying on others to execute the

strategy can stop after Chapter 3 and not feel as if they are missing anything. That said, Parts II and III contain some excellent real-life case studies that also can be read as stand-alone stories. We therefore encourage everyone to thumb through the book and select topics or cases that are of greatest interest.



CHAPTER ONE

What Is Lean and Why Is It Important?

Lean creates value. And it does that by creating competitive advantages that better satisfy the customer.

Joe Stenzel¹

Lean Integration is not a one-time effort; you can't just flip a switch and proclaim to be done. It is a long-term strategy for how an organization approaches the challenges of process and data integration. Lean can and does deliver early benefits, but it doesn't end there. Lean principles such as waste elimination are never-ending activities that result in ongoing benefits. Furthermore, some Lean objectives such as becoming a team-based learning organization with a sustainable culture of continuous improvement may require years to change entrenched bad habits.

Before you start on the Lean journey, therefore, you should be clear about why you are doing so. This chapter, and the rest of the book, will elaborate on the technical merits and business value of Lean Integration and how to

1. Joe Stenzel, *Lean Accounting: Best Practices for Sustainable Integration* (John Wiley & Sons, 2007), Kindle loc. 1317–18.

implement a program that delivers on the promise. Here is a summary of why you would want to:

- **Efficiency:** Lean Integration teams typically realize 50 percent labor productivity improvements and 90 percent lead-time reduction through value stream mapping and continuous efforts to eliminate non-value-added activities. The continuous improvement case study (Chapter 5) is an excellent example.
- **Agility:** Take integration off the critical path on projects by using highly automated processes, reusable components, and self-service delivery models. The mass customization case study (Chapter 8) demonstrates key elements of this benefit.
- **Data quality:** Establish one version of the truth by treating data as an asset, establishing effective information models, and engaging business leaders and front-line staff to accept accountability for data quality. The Smith & Nephew case study (Chapter 6) shows how this is possible.
- **Governance:** Measure the business value of integration by establishing metrics that drive continuous improvement, enable benchmarking against market prices, and support regulatory and compliance enforcement. The integration hub case study (Chapter 10) is an excellent example of effective data governance.
- **Innovation:** Enable staff to innovate and test new ideas by using fact-based problem solving and automating “routine” integration tasks to give staff more time for value-added activities. The Wells Fargo business process automation case study (Chapter 9) is a compelling example of automation enabling innovation.
- **Staff morale:** Increase the engagement and motivation of IT staff by empowering cross-functional teams to drive bottom-up improvements. The decentralized enterprise case study (Chapter 12) shows how staff can be engaged and work together across highly independent business units.

Achieving all these benefits will take time, but we hope that after you have finished reading this book, you will agree with us that these benefits are real and achievable. Most important, we hope that you will have learned enough to start the Lean journey with confidence.

Let’s start by exploring one of the major challenges in most non-Lean IT organizations: the rapid pace of change and surviving at the edge of chaos.

Constant Rapid Change and Organizational Agility

Much has been written about the accelerating pace of change in the global business environment and the exponential growth in IT systems and data. While rapid change is the modern enterprise reality, the question is how organizations can manage the changes. At one end of the spectrum we find agile data-driven organizations that are able to quickly adapt to market opportunities and regulatory demands, leverage business intelligence for competitive advantage, and regularly invest in simplification to stay ahead of the IT complexity wave. At the other end of the spectrum we find organizations that operate at the edge of chaos, constantly fighting fires and barely in control of a constantly changing environment. You may be somewhere in the middle, but on balance we find more organizations at the edge of chaos rather than at the agile data-driven end of the spectrum.

Here is a quick test you can perform to determine if your IT organization is at the edge of chaos. Look up a few of the major production incidents that have occurred in the past year and that have been closely analyzed and well documented. If there haven't been any, that might be a sign that you are *not* on the edge of chaos (unless your organization has a culture of firefighting without postmortems). Assuming you have a few, how many findings are documented for each production incident? Are there one or two issues that contributed to the outage, or are there dozens of findings and follow-up action items?

We're not talking about the root cause of the incident. As a general rule, an analysis of most production incidents results in identifying a single, and often very simple, failure that caused a chain reaction of events resulting in a major outage. But we also find that for virtually all major outages there is a host of contributing factors that delayed the recovery process or amplified the impact.

Here is a typical example: An air conditioner fails, the backup air conditioner fails as well, the room overheats, the lights-out data center sends an automatic page to the night operator, the pager battery is dead, a disk controller fails when it overheats, the failure shuts down a batch update application, a dependent application is put on hold waiting for the first one to complete, an automatic page to the application owner is sent out once the service level agreement (SLA) for the required completion is missed, the application owner quit a month ago and the new owner's pager has not been

updated in the phone list, the chain reaction sets off dozens of alarms, and a major outage is declared which triggers 30 staff members to dial into the recovery bridge line, the volume of alarms creates conflicting information about the cause of the problem which delays problem analysis for several hours, and so on and so on.

Based on our experience with hundreds of similar incidents in banks, retail organizations, manufacturers, telecommunications companies, health care providers, utilities, and government agencies, we have made two key observations: (1) There is never just one thing that contributes to a major outage, and (2) the exact same combination of factors never happens twice. The pattern is that there is no pattern—which is a good definition of chaos. Our conclusion is that at any given point in time, every large IT organization has hundreds or thousands of undiscovered defects, and all it takes is just the right one to begin a chain reaction that results in a severity 1 outage.

So what does this have to do with Lean? Production failures are examples of the necessity of detecting and dealing with every small problem because it is impossible to predict how they line up to create a catastrophe. Three Mile Island is a classic example. Lean organizations relentlessly improve in numerous small steps. A metaphor for how Lean organizations uncover their problems is to imagine a lake with a rocky bottom, where the rocks represent the many quality and process problems affecting their ability to build the best products for their customers. Metaphorically, they “lower the water level” (reduce their inventories, reduce their batch sizes, and speed up reconfiguring their assembly lines, among other techniques) in order to expose the rocks on the bottom of the lake. Once the “rocks” are exposed, they can focus on continually improving themselves by fixing these problems. Integration systems benefit from “lowering the water level” as well. Every failure of a system uncovers a lack of knowledge about the process or its connections. Problem solving is learning more deeply about our processes, infrastructure, and information domains.

We are of the opinion that the edge of chaos is the normal state of affairs and cannot be mitigated purely by technology. The very nature of systems-of-systems is that they emerge and evolve without a complete (100 percent) understanding of all dependencies and behaviors. There are literally billions of permutations and combinations of the internal states of each software component in a large enterprise, and they are constantly changing. It is virtually impossible to test all of them or to build systems

that can guard against all possible failures. The challenge is stated best in remarks by Fred Brooks in *The Mythical Man-Month*: “Software entities are more complex for their size than perhaps any other human construct, because no two parts are alike. . . . If they are, we make the two similar parts into one, a subroutine.” And “Software systems have orders of magnitude more states than computers do.”²

So what *is* the solution? The solution is to perform IT practices such as integration, change management, enterprise architecture, and project management in a disciplined fashion. Note that discipline is not simply a matter of people doing what they are supposed to do. Lack of discipline is not their problem; it is the problem of their managers who have not ensured that the work process makes failure obvious or who have not trained people to respond to revealed failures first with immediate containment and then with effective countermeasures using PDCA (Plan, Do, Check, and Act).

To effectively counter the effects of chaos, you need to approach integration as an enterprise strategy and not as an ad hoc or project activity. If you view integration as a series of discrete and separate activities that are not connected, you won’t buy into the Lean concept. By virtue of the fact that you are reading this book, the chances are you are among the majority of IT professionals who understand the need for efficiency and the value of reuse and repeatability. After all, we know what happens when you execute project after project without a standard platform and without an integration strategy; 100 percent of the time the result is an integration hairball. There are no counterexamples. When you allow independent project teams to choose their own tools and to apply their own coding, naming, and documentation standards, you eventually end up with a hairball—every time. The hairball is characterized by an overly complex collection of dependencies between application components that is hard to change, expensive to maintain, and unpredictable in operation.

If for whatever reason you remain fixed in the paradigm that integration is a project process as opposed to an ongoing process, there are many methodologies to choose from. Virtually all large consulting firms have a proprietary methodology that they would be happy to share with you if you hire them,

2. Frederick P. Brooks, Jr., *The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition* (Addison-Wesley, 2004), pp. 182, 183.

and some of them will even sell it to you. Some integration platform suppliers make their integration methodology available to customers at no cost.

But if you perceive the integration challenge to be more than a project activity—in other words, an ongoing, sustainable discipline—you need another approach. Some alternatives that you may consider are IT service management practices such as ITIL (Information Technology Infrastructure Library), IT governance practices such as COBIT (Control Objectives for Information and Technology), IT architecture practices such as TOGAF (The Open Group Architecture Framework), software engineering practices such as CMM (Capability Maturity Model), or generalized quality management practices such as Six Sigma. All of these are well-established management systems that inherently, because of their holistic enterprise-wide perspective, provide a measure of sustainable integration. That said, none of them provides detailed practices for sustaining solutions to data quality or integration issues that emerge from information exchanges between independently managed applications, with incompatible data models that evolve independently. In short, these “off the shelf” methods aren’t sustainable since they are not your own. Different business contexts, service sets, products, and corporate cultures need different practices. Every enterprise ultimately needs to grow its own methods and practices, drawing from the principles of Lean Integration.

Another alternative to fixing the hairball issue that is often considered is the enterprise resource planning (ERP) architecture, a monolithic integrated application. The rationale for this argument is that you can make the integration problem go away by simply buying all the software from one vendor. In practice this approach doesn’t work except in very unique situations such as in an organization that has a narrow business scope and a rigid operating model, is prepared to accept the trade-off of simply “doing without” if the chosen software package doesn’t offer a solution, and is resigned to not growing or getting involved in any mergers or acquisitions. This combination of circumstances is rare in the modern business economy. The reality is that the complexity of most enterprises, and the variation in business processes, simply cannot be handled by one software application.

A final alternative that some organizations consider is to outsource the entire IT department. This doesn’t actually solve the integration challenges; it simply transfers them to someone else. In some respects outsourcing can make the problem worse since integration is not simply an IT problem; it is a problem of alignment across business functions. In an outsourced business model,

the formality of the arrangement between the company and the supplier may handcuff the mutual collaboration that is generally necessary for a sustainable integration scenario. On the other hand, if you outsource your IT function, you may insist (contractually) that the supplier provide a sustainable approach to integration. In this case you may want to ask your supplier to read this book and then write the principles of Lean Integration into the contract.

In summary, Lean transforms integration from an art into a science, a repeatable and teachable methodology that shifts the focus from integration as a point-in-time activity to integration as a sustainable activity that enables organizational agility. This is perhaps the greatest value of Lean Integration—the ability of the business to change rapidly without compromising on IT risk or quality, in other words, transforming the organization from one on the edge of chaos into an agile data-driven enterprise.

The Case for Lean Integration

The edge of chaos discussion makes the case for Lean Integration from a practitioner's perspective, that is, that technology alone cannot solve the problem of complexity and that other disciplines are required. But that is more of an intellectual response to the challenge and still leaves the five questions we posed in the Introduction unanswered. Let's address them now.

“Why Lean?” and “So What?”

In financial terms, the value of Lean comes from two sources: economies of scale and reduction in variation. Development of data and process integration points is a manufacturing process. We know from years of research in manufacturing that every time you double volume, costs drop by 15 to 25 percent.³ There is a point of diminishing returns since it becomes harder and harder to double volume, but it doesn't take too many doublings to realize an order-of-magnitude reduction in cost. Second, we also know that manufacturing production costs increase from 25 to 35 percent each time variation doubles. The degree of integration variation today in many organizations is staggering in terms of both the variety of tools that are used and the variety of

3. George Stalk, “Time—The Next Source of Competitive Advantage,” *Harvard Business Review*, no. 4 (July–August 1988).

standards that are applied to their implementation. That is why most organizations have a hairball—thousands of integrations that are “works of art.”

Some studies by various analyst firms have pegged the cost of integration at 50 to 70 percent of the IT budget. This is huge! Lean Integration achieves both economies of scale and reduction in variation to reduce integration costs by 25 percent or more. This book explores some specific case studies that we hope will convince you that not only are these cost savings real, but you can realize them in your organization as well.

“As a Business Executive, What Problems Will It Help Me Solve?”

The answer is different for various stakeholders. For IT professionals, the biggest reason is to do more with less. Budgets are constantly being cut while expectations of what IT can deliver are rising; Lean is a great way to respond because it embodies continuous improvement principles so that you can keep cutting your costs every year. By doing so, you get to keep your job and not be outsourced or displaced by a third party.

For a line-of-business owner, the big problems Lean addresses are time to market and top-line revenue growth by acquiring and keeping customers. A Lean organization can deliver solutions faster (just in time) because of automated processes and mass customization methods that are supported by the technology of an Integration Factory. And an integrated environment drives revenue growth through more effective use of holistic information, better management decisions, and improved customer experiences.

For an enterprise owner, the biggest reasons for a Lean Integration strategy include alignment, governance, regulatory compliance, and risk reduction. All of these are powerful incentives, but alignment across functions and business units may be the strongest contributor to sustained competitive advantage. By simply stopping the disagreement across teams, organizations can solve problems faster than the competition.

In summary, Lean Integration helps to reduce costs, shorten time to market, increase revenue, strengthen governance, and provide a sustainable competitive advantage. If this sounds too good to be true, we ask you to reserve judgment until you finish reading all the case studies and detailed “how-to” practices. One word of caution about the case studies: They convey how example organizations solved their problems in their context. The same practices may not apply to your organization in exactly the same way, but the thinking that went into them and the patterns may well do so.

“As an IT Leader or Line-of-Business Owner, Why Am I Going to Make a Considerable Investment in Lean Integration?”

Get results faster—and be able to sustain them in operation. Lean is about lead-time reduction, quality improvements, and cost reduction. Lean delivers results faster because it focuses heavily on techniques that deliver only what the customer needs (no “gold-plating”): process automation and mass customization. In terms of ongoing operations, Lean is a data-driven, fact-based methodology that relies heavily on metrics to ensure that quality and performance remain at a high level.

“How Is This Different from Other Methods, Approaches, and Frameworks?”

Two words: *sustainable* and *holistic*. Other integration approaches either tackle only a part of the problem or tackle the problem only for the short term at a point in time. The predominant integration strategy, even today, is customized hand-coded solutions on a project-by-project basis without a master plan. The result is many “works of art” in production that are expensive to maintain, require a long time to change, and are brittle in operation.

Note that Chapter 12, Integration Methodology, includes a section that explicitly compares agile and Lean methodologies.

“Why Am I as an IT Professional Going to Embrace and Sell Lean Integration Internally?”

Because it will advance your career. Time and time again we have seen successful integration teams grow from a handful of staff to 100 or more, not because of a power grab, but because of scope increases at the direction of senior management. Successful team members become managers, and managers become directors or move into other functions in the enterprise in order to address other cross-functional business needs. In short, Lean Integration is about effective execution, which is a highly valued business skill.

What Is Integration?

This seems like a simple question, but what exactly is integration? Here are a couple of examples of how others have answered this question:

- “The extent to which various information systems are formally linked for sharing of consistent information within an enterprise”⁴
- “Seamless integration of all information flowing through a company”⁵

A careful review of these and other definitions reveals a number of common themes that describe the essence of integration:

- **Formally linked:** Communication between systems is standardized.
- **Seamless:** Complexities are invisible to end users.
- **Coordinated manner:** Communication is disciplined.
- **Synergy:** Value is added to the enterprise.
- **Single transactions that spawn multiple updates:** There is master management of data.
- **Fewer number of systems:** The cost of ownership is lower.
- **Non-duplicated data:** The focus is on eliminating redundancy and reducing costs of management.

The most comprehensive definition of integration we have come across is in Lester Singletary’s doctoral dissertation.⁶ It provides a rich definition by examining the domain of integration from four perspectives: (1) what integration is in terms of its attributes or operational characteristics, (2) the benefits or resultant outcomes of effective integration efforts, (3) the risks or challenges that arise from integrations, and (4) the metrics that provide a measure of objectivity concerning the integrated environment. Figure 1.1 has been adapted from Singletary’s paper and serves as a foundation for our view of integration in this book.

To put all this together, our definition of integration, therefore, is as follows:

Integration: An **infrastructure** for enabling **efficient data sharing** across **incompatible applications** that **evolve independently** in a coordinated manner to serve the needs of the **enterprise** and its **stakeholders**.

4. G. Bhatt, “Enterprise Information Systems Integration and Business Process Improvement Initiative: An Empirical Study,” *Proceedings of the First Conference of the Association for Information Systems (AIS)*, 1995.

5. T. H. Davenport, “Putting the Enterprise into the Enterprise System,” *Harvard Business Review* 16, no. 4 (July–August 1998), pp. 121–31.

6. Lester A. Singletary, “Empirical Study of Attributed and Perceived Benefits of Applications Integration for Enterprise Systems” (PhD dissertation, Louisiana State University, 2003).

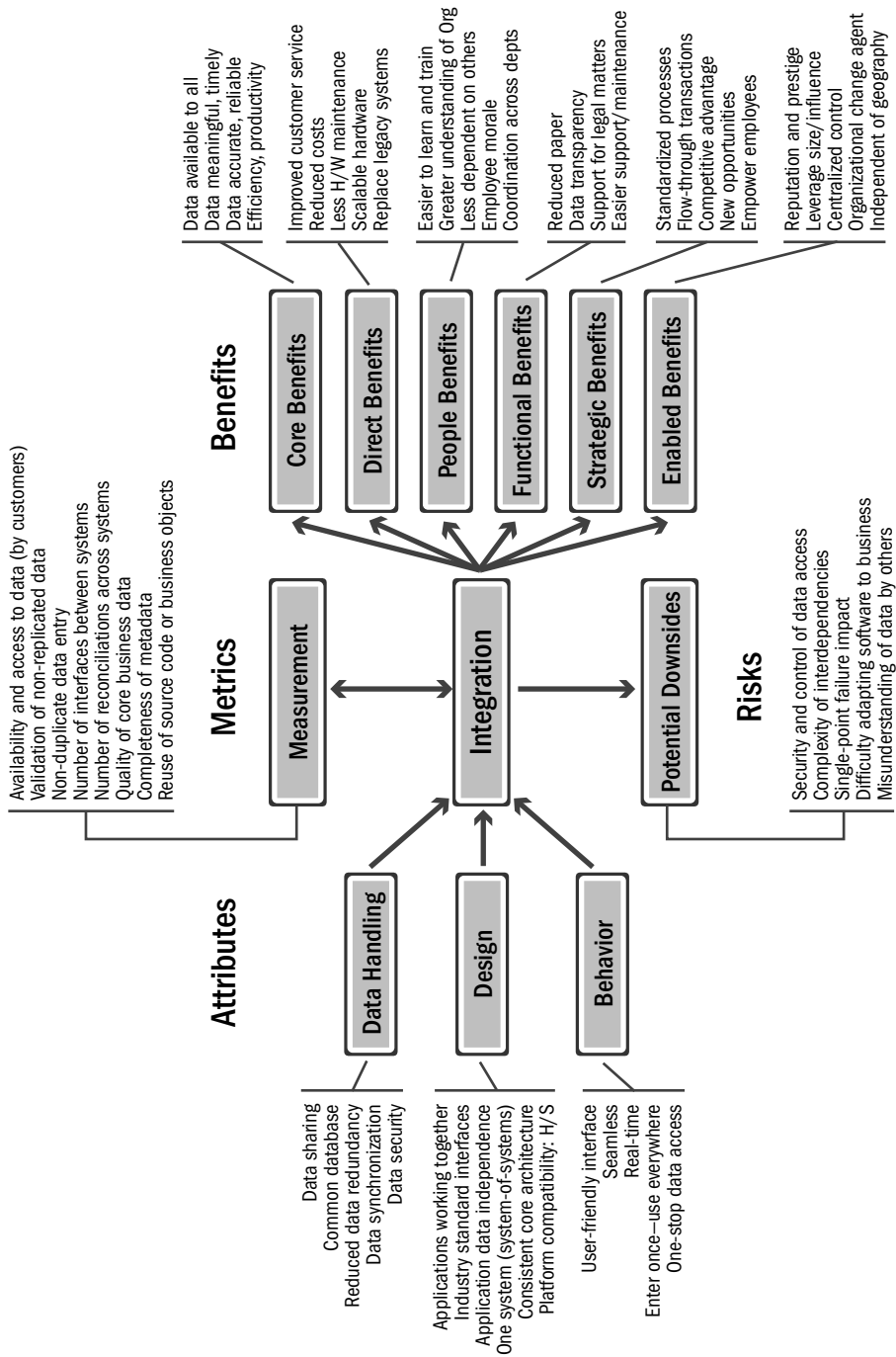


Figure 1.1 Definition of integration

- **Infrastructure:** a combination of people, process, policy, and technology elements that work together (e.g., highway transportation infrastructure)
- **Efficient data sharing:** accessing data and functions from disparate systems without appreciable delay to create a combined and consistent view of core information for use across the organization to improve business decisions and operations
- **Incompatible applications:** systems that are based on different architectures, technology platforms, or data models
- **Evolve independently:** management decisions to change applications (or parts of an application) are made by different organizational groups or external suppliers on independent timelines that are not controlled by a master schedule
- **Enterprise:** the organization unit that is the focus of the integration effort; it could be a department, division, an entire company, or part of a supply chain
- **Stakeholders:** the customers of the enterprise, its owners, and its employees, including management, end users, and IT professionals

Integration Maturity Levels

Another way to look at integration is to examine how integration technologies and management practices have evolved and matured over the past 50 years. Figure 1.2 summarizes four stages of evolution that have contributed to increasingly higher levels of operational efficiency. Hand coding was the only technology available until around 1990 and is still a common practice today, but it is gradually being replaced by modern methods. The movement to standard tools, more commonly known as middleware, began in the 1990s, followed by industry consolidation of tool vendors during the first decade of the 2000s, resulting in more “suites” of tools that provide the foundation for an enterprise integration platform.

As we look to the future, we see the emergence of the Integration Factory as the next wave of integration technology in combination with formal management disciplines. This wave stems from the realization that of the thousands of integration points that are created in an enterprise, the vast majority are incredibly similar to each other in terms of their structure and processing approach. In effect, most integration logic falls into one of a couple of dozen

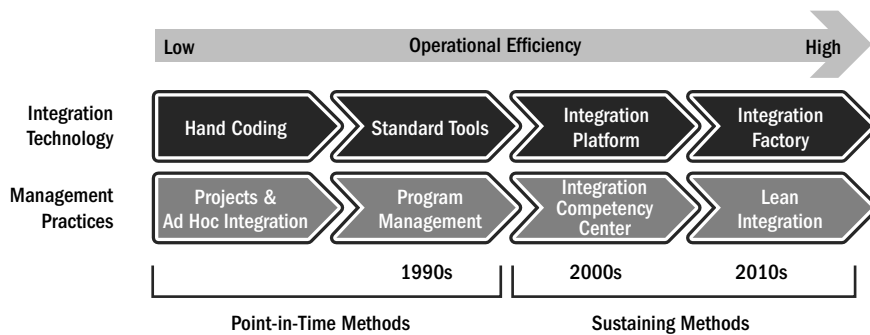


Figure 1.2 Evolution of integration technology and management practices

different “patterns” or “templates,” where the exact data being moved and transformed may be different, but the general flow and error-handling approach are the same. An Integration Factory adds a high degree of automation to the process of building and sustaining integration points. We believe the Integration Factory, described in detail in Chapter 3, will be the dominant new “wave” of middleware for the next decade (2010s).

Management practices have also evolved from ad hoc or point-in-time projects, to broad-based programs (projects of projects), to Integration Competency Centers (ICCs), and now to Lean Integration. A major paradigm shift began early in the current century around the view of integration as a sustaining practice. The first wave of sustainable management practices is encapsulated by the ICC. It focused primarily on standardizing projects, tools, processes, and technology across the enterprise and addressing organizational issues related to shared services and staff competencies. The second wave of sustainable practices is the subject of this book: the application of Lean principles and techniques to eliminate waste, optimize the entire value chain, and continuously improve. The management practice that optimizes the benefits of the Integration Factory is Lean Integration. The combination of factory technologies and Lean practices results in significant and sustainable business benefits.

The timeline shown on the bottom of Figure 1.2 represents the period when the technology and management practices achieved broad-based acceptance. We didn’t put a date on the first evolutionary state since it has been with us since the beginning of the computer software era. The earlier stages of evolution don’t die off with the introduction of a new level of

maturity. In fact, there are times when hand coding for ad hoc integration needs still makes sense today. That said, each stage of evolution borrows lessons from prior stages to improve its efficacy. We predict that Lean practices, in combination with past experiences in project, program, and ICC practices, will become the dominant leading practice around the globe during the 2010s.

In Part III of this book we refer to these four stages of evolutionary maturity when discussing the eight integration competency areas. The shorthand labels we use are as follows:

1. **Project:** disciplines that optimize integration solutions around time and scope boundaries related to a single initiative
2. **Program:** disciplines that optimize integration of specific cross-functional business collaborations, usually through a related collection of projects
3. **Sustaining:** disciplines that optimize information access and controls at the enterprise level and view integration as an ongoing activity independent of projects
4. **Lean:** disciplines that optimize the entire information delivery value chain through continuous improvement driven by all participants in the process

We think of this last level of maturity as self-sustaining once it becomes broadly entrenched in the organization.

We don't spend much time in this book discussing project or program methods since these are mature practices for which a large body of knowledge is available. Our focus is primarily on sustaining practices and how Lean thinking can be applied to achieve the highest levels of efficiency, performance, and effectiveness.

Economies of Scale (the Integration Market)

As stated earlier, the benefits of Lean are economies of scale and reduction in variation. As a general rule, doubling volume reduces costs by 15 to 25 percent, and doubling variation increases costs by 25 to 35 percent. The ideal low-cost model, therefore, is maximum standardization and maximum volume. But how exactly is this accomplished in a Lean Integration context?

A core concept is to view the collection of information exchanges between business applications in an enterprise as a “market” rather than as a bunch of private point-to-point relationships. The predominant integration approach over the past 20 years has been point-to-point integration. In other words, if two business systems need to exchange information, the owners and subject matter experts (SMEs) of the two systems would get together and agree on what information needed to be exchanged, the definition and meaning of the data, the business rules associated with any transformations or filters, the interface specifications, and the transport protocol. If anything needed to change once it was in operation, they would meet again and repeat the same process.

For a small number of systems and a small number of information exchanges, this process is adequate and manageable. The problem with a hand-coded or manual method is that it doesn’t scale, just as manual methods for other processes don’t scale well. Certainly if a second integration point is added to the same two systems, and the same two SMEs work together and use the same protocols, documentation conventions, and so on, the time and cost to build and sustain the integrations will indeed follow the economy of scale cost curve. But in a large enterprise with hundreds or thousands of applications, if each exchange is viewed as strictly an agreement between the immediate two parties, diseconomies begin to creep into the equation from several perspectives.

Imagine trying to follow a flow of financial information from a retail point-of-sale application, to the sales management system (which reconciles sales transactions with refunds, returns, exchanges, and other adjustments), to the inventory management system, to the general ledger system, to the financial reporting system. In a simple scenario, this involves four information exchanges among five systems. If each system uses different development languages, protocols, documentation conventions, interface specifications, and monitoring tools and was developed by different individuals, not only will we *not* receive the benefits from quadrupling volume from one integration to four, but we will in fact increase costs.

This example reflects the two largest factors that drive diseconomies: the cost of communication between teams and duplication of effort. Additional factors can drive further diseconomies, such as the following:

- **Top-heavy organizations:** As organizations get larger and add more layers of management, more and more effort needs to be expended on

integrated solutions that require collaboration and agreement across teams that each play a narrow functional role.

- **Office politics:** Disagreements across teams are a result of different motivations and agendas, usually a result of conflicting goals and metrics but also sometimes caused by the “not invented here” syndrome.
- **Isolation** of decision makers from the results of their decisions: Senior managers may need to make decisions, such as how much of a budget to allocate to a given group, without a clear picture of what the group does and what value it brings to the organization.
- **Slow response time:** Delays are caused by multiple handoffs between teams or by queuing requests for information or support from other groups.
- **Inertia:** People are unwilling to change or are opposed to standards.
- **Cannibalization:** Limited resources (such as SMEs in specific business domains) are consumed for project B, slowing down progress on project A.

The degree of integration variation in many organizations is staggering in terms of both the variety of technology that is used and the variety of standards that are applied to their implementation. That is why most organizations have a hairball—hundreds or thousands of integrations that are “works of art.”

The alternative is to view the need for information exchanges across applications as a market economy, and to serve the market with an efficient shared-services delivery model in order to gain economies of scale. For example, multiple groups within an organization may perform similar activities but do so with varying degrees of efficiency and consistency. Centralizing the activities makes it much easier to standardize and streamline the work, thereby reducing the cost per unit of work while improving the quality and consistency.

The two graphs in Figure 1.3 are borrowed from the field of economics and show the relationships between costs and volumes. These graphs reflect the well-understood laws of diminishing returns and economies of scale. The chart on the left reflects a manual or low-tech operation, such as when information exchanges are developed using custom hand-coded integration solutions. In this scenario, there are few (if any) capital costs since existing enterprise tools such as COBOL, Java, or SQL are used to build the integration. The overall average cost per integration initially falls as the individuals doing the work gain experience and are able to share some of their experience and knowledge, but then at some point it starts to increase as diseconomies

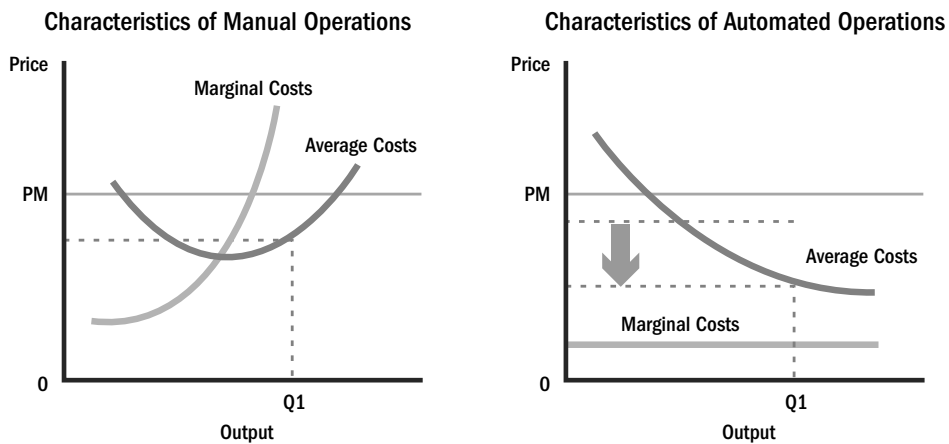


Figure 1.3 Diminishing returns and economies of scale

emerge. In terms of the marginal costs (i.e., the incremental cost for each additional integration), initially the curve is somewhat flat since the first integration developer can develop a second or third integration with a similar effort. The average cost also falls initially since the fixed costs of the developer (hiring costs, office space, desktop PC, etc.) are amortized over more than one integration. As the volume of integrations increases, however, the marginal costs increase on an exponential basis, and the average costs begin to increase as more and more diseconomies emerge from the increasing complexity and number of unique integration points.

The chart on the right of the figure shows the cost curve as a result of a capital investment in tools and infrastructure (such as an Integration Factory) to largely automate and standardize the development effort. Note that in this scenario, the marginal costs are small and constant. For example, it might cost Microsoft \$5 billion to develop a new version of Windows, but once developed, it costs just pennies to make another electronic copy. The marginal cost per copy of Windows is essentially the same whether 1 copy or 1,000 copies are made, but the average cost drops significantly and continuously in this scenario as volume increases and as the up-front fixed costs are amortized over more and more units.

The key challenge for organizations is to determine at what level of integration complexity do diminishing returns begin to emerge from manual hand-coded solutions, and how much capital investment is warranted to

achieve the time and cost advantages of a high-volume Integration Factory. The answer to this will become clearer in Parts II and III, where we discuss the Lean principles related to continuous improvement, mass customization, and process automation, and the financial management competency area.

Getting Started: Incremental Integration without “Boiling the Ocean”

Parts II and III of the book provide detailed and specific advice on how to implement a sustainable Lean Integration practice, but before you dig into the details, it is important to understand the approach options and related prerequisites.

There are two fundamental implementation styles for Lean Integration: top-down and bottom-up. The top-down style starts with an explicit strategy with clearly defined (and measurable) outcomes and is led by top-level executives. The bottom-up style, which is sometimes referred to as a “grassroots” movement, is primarily driven by front-line staff or managers with leadership qualities. The top-down approach generally delivers results more quickly but may be more disruptive. You can think of these styles as revolutionary versus evolutionary. Both are viable.

While Lean Integration is relevant to all large organizations that use information to run their business, there are several prerequisites for a successful Lean journey. The following five questions provide a checklist to see if Lean Integration is appropriate to your organization and which style may be most suitable:

1. Do you have senior executive support for improving how integration problems are solved for the business?

Support from a senior executive in the organization is necessary for getting change started and critical for sustaining continuous improvement initiatives. Ideally the support should come from more than one executive, at a senior level such as the CXO, and it should be “active” support. You want the senior executives to be leading the effort by example, pulling the desired behaviors and patterns of thought from the rest of the organization.

It might be sufficient if the support is from only one executive, and if that person is one level down from C-level, but it gets harder and harder to drive the investments and necessary changes as you water down the

top-level support. The level of executive support should be as big as the opportunity. Even with a bottom-up implementation style, you need some level of executive support or awareness. At some point, if you don't have the support, you are simply not ready to formally tackle a Lean Integration strategy. Instead, just keep doing your assigned job and continue lobbying for top-level support.

2. Do you have a committed practice leader?

The second prerequisite is a committed practice leader. By “committed” we don't mean that the leader needs to be an expert in all the principles and competencies on day one, but the person does need to have the capability to become an expert and should be determined to do so through sustained personal effort. Furthermore, it is ideal if this individual is somewhat entrepreneurial, has a thick skin, is customer-oriented, and has the characteristics of a change agent (see Chapter 6 on team empowerment for more details).

If you don't have top leadership support or a committed practice leader, there is little chance of success. This is not to suggest that a grassroots movement isn't a viable way to get the ball rolling, but at some point the bottom-up movement needs to build support from the top in order to institutionalize the changes that will be necessary to sustain the shift from siloed operations to integrated value chains.

3. Is your “Lean director” an effective change agent?

Having a Lean director who is an effective change agent is slightly different from having one who is “committed.” The Lean champion for an organization may indeed have all the right motivations and intentions but simply have the wrong talents. For example, an integrator needs to be able to check his or her ego at the door when going into a meeting to facilitate a resolution between individuals, who have their own egos. Furthermore, a Lean perspective requires one to think outside the box—in fact, to not even see a box and to think of his or her responsibilities in the broadest possible terms. Refer to the section on Change Agent Leadership in Chapter 6 for a description of essential leadership capabilities.

4. Is your corporate culture receptive to cross-organizational collaboration and cooperation?

Many (maybe even most) organizations have entrenched views of independent functional groups, which is not a showstopper for a Lean program. But

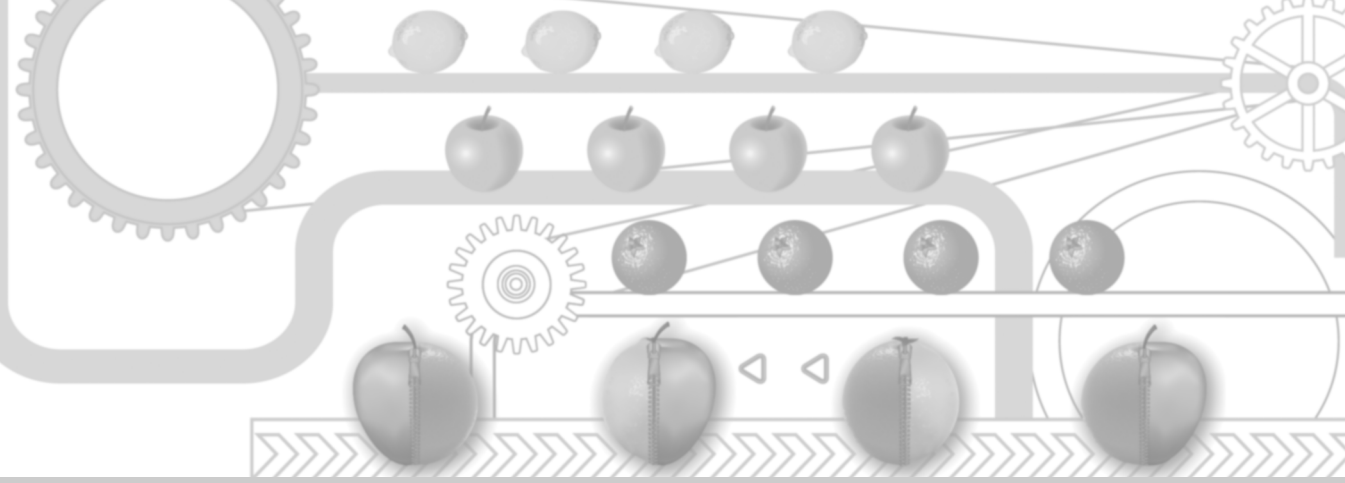
if the culture is one where independence is seen as the source of the organization's success and creativity, and variation is a core element of its strategy, a Lean approach will likely be a futile effort since Lean requires cooperation and collaboration across functional lines. A corporate culture of autonomous functional groups with a strong emphasis on innovation and variation typically has problems implementing Lean thinking.

5. Can your organization take a longer-term view of the business?

A Lean strategy is a long-term strategy. This is not to say that a Lean program can't deliver benefits quickly in the short term—it certainly can. But Lean is ultimately about long-term sustainable practices. Some decisions and investments that will be required need to be made with a long-term payback in mind. If the organization is strictly focused on surviving quarter by quarter and does little or no planning beyond the current fiscal year, a Lean program won't achieve its potential.

If you are in an organizational environment where you answered no to one or more of these questions, and you feel compelled to implement a Lean program, you could try to start a grassroots movement and continue lobbying senior leadership until you find a strong champion. Or you could move to another organization. There are indeed some organizational contexts in which starting a Lean program is the equivalent of banging your head against the wall. We hope this checklist will help you to avoid unnecessary headaches.

Lean requires a holistic implementation strategy or vision, but it can be implemented in incremental steps. In fact, it is virtually impossible to implement it all at once, unless for some reason the CEO decides to assign an entire team with a big budget to fast-track the implementation. The idea is to make Lean Integration a long-term sustainable process. When we say "long-term" we are talking about 10 to 20 years, not just the next few years. When you take a long-term view, your approach changes. It certainly is necessary to have a long-term vision and plan, but it is absolutely acceptable, and in many respects necessary, to implement it incrementally in order to enable organizational learning. In the same way, an ICC can start with a small team and a narrow scope and grow it over time to a broad-based Lean Integration practice through excellent execution and positive business results.



APPENDIX A

Lean Terminology

5S: A Lean methodology to create a productive and safe workplace. It consists of

- **Sort:** Keep only really necessary things.
- **Set in order:** Arrange and store everything to be easy to locate, access, and return to its storage place.
- **Shine:** Keep the entire work area clean and ready for production.
- **Standardize:** Create standard work documentation for tasks needed to keep work areas clean and orderly, with diagrams documenting where all tools and supplies are to be stored.
- **Sustain:** Ensure that workplaces are maintained as documented in standard work instructions through empowering operators to organize, maintain, and improve them; also include areas to be audited as a part of layered audit systems.

5 Whys: A method for identifying problem root causes by repetitively asking why, getting a response, and asking why again. The process continues until the identified cause can be linked back to the original problem and

it is confirmed that removing or controlling the cause would eliminate the problem.

A3: Another Lean tool in common use; it is a persuasive one-page analysis and synthesis of a problem on A3-size paper (approximately 11 × 17 inches, hence the name), intended to get people involved and to act in ways they otherwise wouldn't. A3 is used effectively for a number of purposes, including annual plan goal deployment, project management, and problem solving. The content should

- Provide the background, context, and importance of the problem
- Assess current performance and the gap versus standard and perfection
- Determine specific goals and objectives of the project
- Analyze the problem to identify its root causes
- Recommend countermeasures that will eliminate defined root causes
- Define an action plan to implement recommended countermeasures
- Establish a measurement, risk mitigation, and review process and standardization plan

There are many benefits of organization-wide A3 use, among them:

- A3's standardized presentation format is understood across the organization so energy is focused on solving problems.
- There is increased speed of problem solving as repetitive use builds stronger problem-solving skills.
- A3s generate ideas from the entire organization as they are posted in areas accessible to everyone.
- They tell shop floor associates they are full team members with management.

Error-proofing—*poka-yoke*: A Lean set of practices intended to prevent the existence of conditions that can result in defects, or if it is not practical to prevent defects, to catch them as close to their source as possible.

***Jidoka*:** Providing machines and operators with the ability to detect when abnormal conditions have occurred and immediately stop work. *Jidoka* is sometimes called “autonomation,” meaning automation with human intelligence.

Just-in-time (JIT) production: A production system of making and delivering just what is needed, just when it is needed, and in just the amount needed.

***Kaizen*:** The continuous improvement of a value stream or an individual process to create more value with less waste. There are two levels of *kaizen*: System or flow *kaizen* focuses on the overall value stream, and process *kaizen* focuses on individual processes.

***Kaizen events*:** A short-duration improvement project, normally one day to one week, completed by a team following a PDCA-based event methodology.

***Kanban*:** A Japanese term for “sign” or “signboard.” In Lean it is a signaling device giving authorization and instructions for producing or withdrawing (conveyance) items in a pull system.

Layered audit system: A system of auditing work areas involving operators and all levels of leadership. Shop floor operators audit their areas every day, team leaders may audit their areas weekly, area managers monthly, and plant managers maybe semiannually. Its purpose is compliance with standard work, coaching, and identification of future opportunities to improve.

Lean production system: A business system for organizing and managing product development, operations, suppliers, and customer relations that requires less human effort, less space, less capital, and less time to make products with fewer defects to precise customer desires, compared with the previous system of mass production. The term was coined by John Krafcik, a research assistant at MIT with the International Motor Vehicle Program in the late 1980s.

Level scheduling: A scheduling method of minimizing the effects of mix and volume variability. Work is released to production by pitch, a volume of product defined based on an increment of time to standardize small lot production. It functions in combination with shortening process setup times so smaller lots can be run efficiently. Production volume flexibility is created by using cell design and standard work to be able to quickly flex crew sizes based on Takt time.

Management by fact (MBF): A tool used in a number of methodologies including Six Sigma and CMM. It also has many similarities to the A3 problem-solving process. It is a concise summary of quantified problem statement, performance history, prioritized root causes, and corresponding countermeasures for the purpose of data-driven problem analysis

and management. MBF uses “4 Whats” to help quantify the problem statement and the gap between actual and desired performance, and “5 Whys” to determine root causes. (See Figure A.1.)

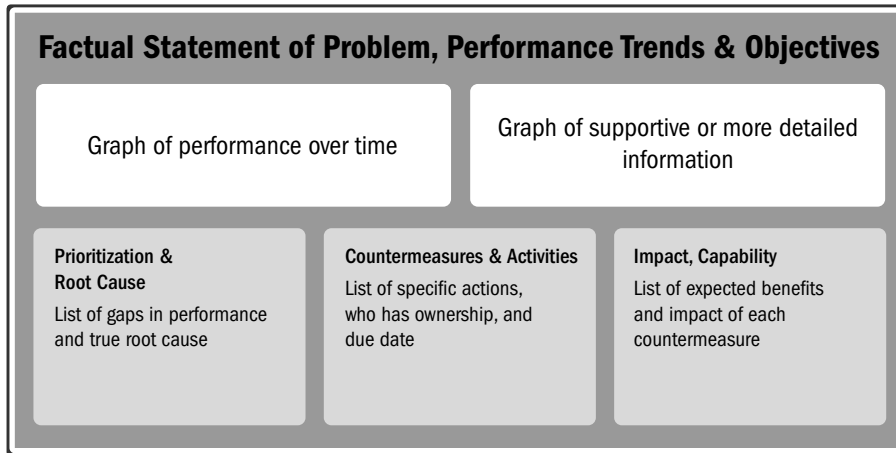


Figure A.1 MBF template

Obeya: *Obeya* in Japanese means simply “big room.” It is a major project management tool used to enhance effective, timely communication and ensure project success by shortening the PDCA cycle. Similar in concept to traditional “war rooms,” an *obeya* contains highly visual charts and graphs depicting program timing, milestones, and progress to date and countermeasures to existing timing or technical problems. Project leaders may have desks in the *obeya* to facilitate communication.

Pacemaker process: The production process nearest the customer, where products take on their definition. It sets the pace for a value stream. (The pacemaker process should not be confused with a bottleneck process, which necessarily constrains downstream processes because of a lack of capacity.)

PDCA (Plan, Do, Check, Act): An iterative four-step problem-solving process to:

- Identify the real problem, using the five whys, and define countermeasures to eliminate or control the root causes
- Implement the countermeasures
- Measure the new processes and compare against expected results
- Analyze the differences to determine their cause and repeat the problem-solving cycle if necessary

Process efficiency: The percent of total value stream lead time or throughput time that is value-added; it is calculated by dividing value-added time by total time from receipt of raw materials to shipment to customers.

Production lead time (throughput time and total product cycle time): The time required for a product to move all the way through a process from start to finish. At the plant level this is often termed *door-to-door time*. The concept can also be applied to the time required for a design to progress from start to finish in product development or for a product to proceed from raw materials all the way to the customer.

Source quality: A quality management approach to focus on defect prevention at its source or detection and correction as close to the generating process as possible. Defects are never allowed to be passed on to the next downstream process.

Supermarket: Lean term for an inventory buffer of raw materials, work in process, or finished goods, ensuring reliable supply to a downstream process; created to mitigate the consequences of supply variability, demand variability, and differences in the process times of two sequential processes. Withdrawal of material is authorized by a withdrawal *kanban* from the using process, and replenishment is authorized by a production *kanban* sent to the producing process.

Standard work for managers: Every team member has well-defined standardized work, which is critical to accountability, commitment, and achievement of results through waste elimination. Standardized work is defined for every level of management. Thus, managers must have structured time in their daily schedules for time on the shop floor. The daily schedule must be arranged so that managers review all operations over a defined period. The purposes are to

1. Audit standardized work
2. Coach team members in the Lean system and mentor continuous improvement
3. Follow up on previously identified deviations to ensure corrective action is being completed
4. Identify the next level of system improvement

This is a critical management activity required for sustaining Lean systems. A second standardized work activity for managers is leading policy

deployment, or *hoshin kanri*, to ensure that operations are completely aligned with the annual business plan. The third manager's standardized work activity is conducting regular Lean operational reviews. Regular reviews are required to assure organizational accountability to fulfill operational plan goals.

Takt time: The increment of time, in seconds or minutes, required to produce one unit of product. It is calculated by dividing available production time by customer demand. For example, if a widget factory operates 480 minutes per day and customers demand 240 widgets per day, Takt time is 2 minutes. The purpose of Takt time is to precisely match production with demand. It provides the heartbeat of a Lean production system. The term is German for a precise interval of time such as a musical meter.

Toyota Production System (TPS): The production system developed by Toyota Motor Corporation to provide best quality, lowest cost, and shortest lead time through the elimination of waste. Development of TPS is credited to Taiichi Ohno, Toyota's chief of production in the post–World War II period.

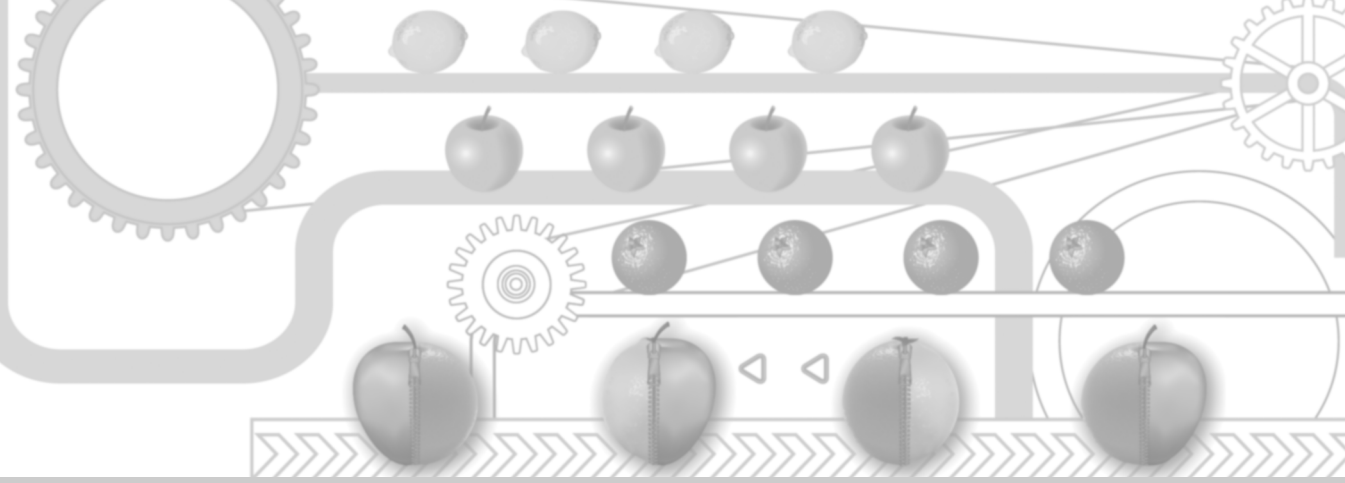
TPM (total productive maintenance): A system for managing facility and machine maintenance focused on elimination of emergency maintenance through rigorous application of preventive and predictive maintenance practices.

Value stream: All of the actions, both value-creating and non-value-creating, required to bring a product from concept to launch and from order to delivery. These include actions to process information from the customer and actions to transform the product on its way to the customer.

Visual management: An important Lean principle and practice. As a principle it is the foundation of making the system transparent so that all can see and understand it, not just the experts. As a practice it means implementing practices, progress metric status, and project implementation status in simple standardized approaches that everyone in the organization can understand. For example, inventory in a supermarket should never fall outside defined minimum or maximum levels. Storage locations in supermarkets are designed to hold the maximum planned inventory. If inventory exceeds the maximum, it must be stored outside designated locations, creating a visible variation that will be caught during the daily audits. These obvious deviations are easily observed and demand immediate follow-up to understand root causes of system failures.

Work cell design: The process of organizing production process steps into logical groups, creating a layout to facilitate minimum materials handling and the most efficient use of operator time. The work at each workstation is balanced to minimize waste and improve flow, resulting in shorter cycle times and lower cost.

Waste: A resource-consuming activity that does not create customer value. In Lean there are seven specific types of waste: excess transportation, inventory, motion, waiting, overproduction, overprocessing, and defects.



APPENDIX B

Integration Laws¹

Integration “laws” are ways of thinking about the fundamental drivers of integration. The laws reflect the reality of dealing with “systems-of-systems” (or complex systems) that are characteristic of enterprise integration. They represent the reality of “what is” rather than “what could be” and, just like the laws of physics, describe many characteristics of the real world.

An effective integration approach must not conflict with the integration laws. Although challenging them won’t land you in jail, ignoring them will likely add to the list of failed integration projects. As you start down the path of your ICC implementation, remember these five laws.

Law #1: The Whole Is Greater than the Sum of Its Parts

The notion of “process decomposition” is deeply ingrained in most analysis techniques used in modern software development life-cycle methodologies. It is based on the presumption that there are natural boundaries along which

1. John G. Schmidt and David Lyle, *Integration Competency Center: An Implementation Methodology* (Informatica Corporation, 2005), pp. 12–14.

to divide a complex system into smaller components for integration. This approach comes from the reductionist perspective, dealing with one dimension of problem analysis.

While this approach helps with tackling relatively simple problems in short time frames, it fails as system complexity increases and natural boundaries disappear. All of the gains achieved by breaking down the big problem are lost as the cost of integrating the small solutions becomes untenable.

Most methodologies fail to realize that the essence of an end-to-end system cannot be captured by studying its individual components alone, and they fail to assign responsibility for the holistic solution. Or if accountability is clear for the initial *construction* of a solution, the solution can deteriorate if no one is responsible for sustaining the end-to-end processes on an ongoing basis.

Law #2: There Is No End State

Organizational entities split, merge, and morph into new structures. Political motivations and boundaries change. Technology evolves, and today's leading edge is tomorrow's legacy. An effective ICC approach must consider the full life cycle of a system and be based on best practices that recognize the adaptive nature of complex systems. From the start, we must plan for constant change.

Furthermore, ICCs must deal with legacy systems based on prior generations of technology. There have been many waves of application technology over the years that seem to move in regular seven-year cycles (e.g., mainframe to mini to microcomputers, monolithic to client/server to Web service applications, etc.). The shift from one wave to the next is neither instantaneous nor is it necessarily economically justified. In fact, a given technology usually lasts through several waves before it is fully replaced. Therefore, the ICC must deal with three to four generations of technology simultaneously.

Law #3: There Are No Universal Standards

Having too many software standards has the same effect as having no standards at all. Even successful standards (such as TCP/IP for the Internet) are not universal. When it comes to software standards such as COBOL or Java, interoperability and transportability come at the expense of vendor-specific extensions, forcing developers to use a less-than-ideal core set of "pure" language features.

The ICC should strive to define and adopt standards within the enterprise, but also work externally with standards organizations to gain agreement

across the industry. That said, the ICC must deal with the reality that many forces—including competition, the “not invented here” syndrome, and evolving technologies—will result in many different standards for the foreseeable future.

Law #4: Information Adapts to Meet Local Needs

The information engineering movement of the early 1990s was based on the incorrect notion that an enterprise can have a single consistent data model without redundancy. A more accurate way to look at information is as follows:

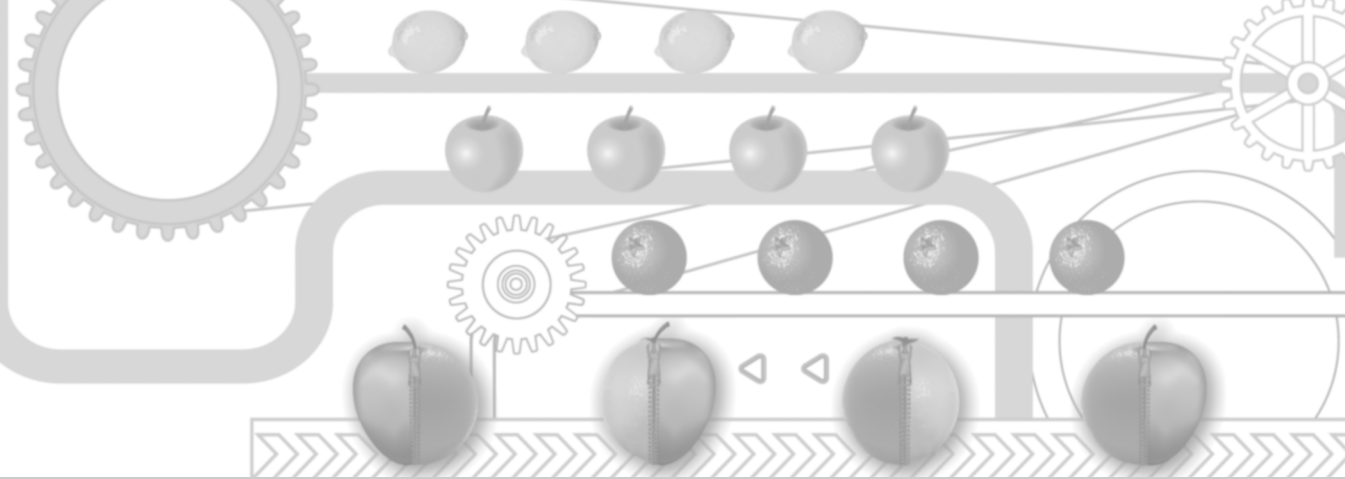
$$\text{Information} = \text{Data} + \text{Context}$$

This formula says that the same data across different domains may have different meanings. For example, a simple attribute such as “Current Customer” can mean something different to the marketing, customer service, and legal departments. An extreme example is gender, which you might think could have only two states: male or female; but one particular enterprise has defined eight different genders. The same thing happens with natural languages (the various meanings that words adopt in different communities). The ICC must embrace informational diversity, recognizing that variations exist, and use techniques to compensate for them.

Law #5: All Details Are Relevant

Abstraction is the practice of representing a problem without all the details, developing a model solution based on the abstract problem, and then using the model to create the real-life solution. The success of this approach depends on our ability to build and use abstract models to manage and direct activities. But the effectiveness of an abstract model is inversely proportional to the complexity of the context, because no details can be safely ignored. The cost of developing and maintaining abstract models of the system and the project can become an economic black hole, consuming all benefits.

A successful ICC deals with this conundrum by decomposing the problem while maintaining a view of the entire picture. Although there is no easy solution, an effective ICC must strive to achieve *dynamic* models—models that are connected to the real world in such a way that when one changes, so does the other. Only then can we attain a truly sustainable integration infrastructure.



APPENDIX C

Glossary

This glossary is divided into two parts: common acronyms, and the most important integration and information technology terms used in this book. Note that this glossary will be maintained on the book's Web site, www.integrationfactory.com.

Note: Italicized acronyms are also included in the following definitions section.

Common Acronyms

| | |
|--------------|--|
| API: | <i>Application program interface</i> |
| B2B: | Business to business |
| B2C: | Business to consumer |
| BI: | Business intelligence |
| BPEL: | Business Process Execution Language |
| BPM: | Business process management or business process modeling |

| | |
|------------------|---|
| BPMN: | Business Process Modeling Notation |
| CDC: | Changed data capture |
| CORBA: | Common Object Request Broker Architecture |
| DBMS: | Database management system |
| DI: | <i>Data integration</i> |
| EAI: | <i>Enterprise application integration</i> |
| EDA: | <i>Event-driven architecture</i> |
| EII: | Enterprise information integration |
| ESB: | <i>Enterprise service bus</i> |
| ETL: | Extract, transform, load |
| FTE: | Full-time equivalent (staff) |
| HTTP: | Hypertext Transfer Protocol |
| ICC: | <i>Integration Competency Center</i> |
| IS: | Information systems (organizational unit) |
| IT: | Information technology (organizational unit) |
| ME&C: | Mutually exclusive and comprehensive |
| MDM: | Master data management |
| MOM: | Message-oriented middleware |
| ODS: | <i>Operational data store</i> |
| POC: | Proof of concept |
| REST: | Representational state transfer |
| SLA: | <i>Service level agreement</i> |
| SOA: | <i>Service-oriented architecture</i> |
| SOAP: | Simple Object Access Protocol |
| SOR: | <i>System of record</i> |
| SQL: | Structured Query Language |
| UDDI: | Universal Description, Discovery, and Integration |
| XML: | eXtensible Markup Language |

Definitions

Application: A deployed and operational IT system that supports business functions and services. Applications encapsulate data and are supported by multiple technology components. Applications may be logical or physical but are distinct from the technology components that are used to construct them.

Application program interface (API): A set of public programmatic interfaces that consists of a language and a message format to communicate with an operating system or other programmatic environment, such as databases, Web servers, and so forth. These messages typically call functions and methods available for application development.

Bus: An abstract software pattern used to transfer data between multiple systems. In contrast to the hub-and-spoke pattern, it uses a federation of components that all follow a common policy or protocol to send, route, and receive messages.

Business glossary: A list of business data elements with a corresponding description, enterprise-level or domain-specific synonyms, validation rules, and other relevant metadata. Used to identify the source of a record, quality metrics, ownership authority, and stewardship responsibility.

Business object: An encapsulated unit of application functionality closely aligned to data and data access considerations.

Business process: A structured description of the activities or tasks that have to be done to fulfill a certain business need. The activities or tasks might be manual steps (human interaction) or automated steps (IT steps). Business processes might be managed and implemented using modeling notations such as BPMN or EPC or execution languages such as BPEL. Some people differentiate between workflows and business processes by stating that business processes describe more generally what has to be done, whereas workflows describe how activities or tasks should be carried out.

Canonical: Reduced to the simplest and most significant form possible without loss of generality.

Canonical data model: The definition of a standard organization view of a particular information subject. To be practical, canonical data models include a mapping back to each application view of the same subject. The canonical data model is frequently implemented as an XML hierarchy. Specific uses include delivering enterprise-wide business intelligence (BI), defining a common view within a service-oriented architecture (SOA), and streamlining software interfaces.

Canonical model: A design pattern used to communicate between different data formats. The basic idea is that rather than writing translators between each and every format (with potential for a combinatorial

explosion), it is sufficient just to write a translator between each format and the canonical format.

Capability: A thing that an organization, person, or system is able to do. Capabilities are typically very coarse-grained and may bring together a combination of people, processes, and technology.

Coupling: A dependency between two computer hardware or software elements or systems. Tight coupling is when the elements are strongly dependent on each other so that when one changes, the other is impacted. Loose coupling reduces dependencies between systems through techniques such as middleware abstraction layers, asynchronous communication, or compensating transactions rather than two-phase commit transactions to maintain consistency. In general, loose coupling leads to more complexity. For this reason, in a specific architectural style it is important find the right amount of loose coupling. Coupling is usually contrasted with cohesion. Low coupling often correlates with high cohesion, and vice versa.

Data entity: An encapsulation of data that is recognized by a business domain expert as a thing. Logical data entities can be tied to applications, repositories, and services and may be structured according to implementation considerations.

Data governance: The policies and processes that continually work to improve and ensure the availability, accessibility, quality, consistency, auditability, and security of data in a company or institution.

Data integration: Accessing data and functions from disparate systems to create a combined and consistent view of core information for use across the organization to improve business decisions and operations.

Data mart: A database structured for specific analysis and historical reporting needs.

Data warehouse: A subject-oriented, integrated, time-variant, and historical collection of summary and detailed data used to support the decision-making and other reporting and analysis needs that require historical, point-in-time information. Data, once captured within the warehouse, is nonvolatile and relevant to a point in time.

Enterprise application integration (EAI): An approach to integrating distributed systems so that they use a common infrastructure (middleware and/or protocol). With this approach, for each system it is enough to

provide and maintain only one adapter to the infrastructure, instead of a specific adapter for each of the systems with which it communicates. The infrastructure might use a bus or hub-and-spoke approach.

Enterprise service bus (ESB): The infrastructure of an SOA landscape that enables the interoperability of services. Its core task is to provide connectivity, data transformations, and routing so that systems can communicate via services.

Event: A notification sent to a more or less well-known set of receivers (consumers). Usually, the receivers of an event have to subscribe to a certain type of event (sent by a certain system or component). Depending on the programming or system model, the systems sending the events (the providers) might or might not know and agree to send the events to the subscribing receivers.

Event-driven architecture (EDA): A software architecture pattern promoting the production, detection, consumption of, and reaction to events.

Governance: The discipline of tracking, managing, and steering an IS/IT landscape. Architectural governance is concerned with change processes (design governance). Operational governance looks at the operational performance of systems against contracted performance levels, the definition of operational performance levels, and the implementation of systems that ensure the effective operation of systems.

Hub and spoke: An abstract software pattern used to transfer data between multiple systems. In contrast to the bus pattern, it uses a central component that coordinates all communication between senders and receivers.

Information object model: A model used to provide traceability from the enterprise function and information subject models to the business glossary (i.e., an information object includes a list of data elements from the business glossary). Possibly used for assessing current information management capabilities (reflected in process and target systems models) or as a conceptual model for custom-developed application components.

Integration: (1) An infrastructure for enabling efficient data sharing across incompatible applications that evolve independently, in a coordinated manner, to serve the needs of the enterprise and its stakeholders. (2) The capability for efficient data sharing across applications in a coordinated manner to serve the needs of the enterprise and its stakeholders. The end result is a system-of-systems that supports sharing data and functions

across incompatible applications to create a combined and consistent view of core information for use across the enterprise to improve business decisions and operations. (3) The capability to constructively face the tensions of incompatible business systems that need to interoperate and, rather than simply coupling them to satisfy a tactical need, generating a creative solution that contains elements of the individual systems but is superior to each—in other words, creating synergy where the whole is greater than the sum of the parts.

Integration Competency Center (ICC): A permanent cross-functional team operating as a shared-services function supporting multiple organizational units and sustaining integration in a coordinated manner. Alternate names include Business Intelligence Competency Center (BICC), Integration Center of Excellence (Integration COE), Data Quality Competency Center (DQCC), SOA Center of Expertise (SOA COE), Integration Solutions Group (ISG), Enterprise Data Management (EDM), and other variants.

Integration Factory: A cohesive integration technology platform that automates the flow of materials and information in the process of building and sustaining integration points. Examples of automation include requirements definition, code generation, testing, and migration of code objects.

Integration point: A data exchange or dependency between two business systems that provides discrete functionality and is designed and managed as a functional unit. An integration point may involve one or more interfaces with each of the systems involved and may involve any number of middleware elements, but it is still considered as one integration point. Conversely, if system A publishes the same data using the same interface to systems B and C, it is considered as two integration points (A–B and A–C). Furthermore, if data is combined from systems X and Y and then sent to system Z as a single message, it is also considered as two integration points (X–Z and Y–Z).

Interface: The externally visible definition of the operations permitted on an application component.

Interoperability: The ability of different systems to communicate with each other. Interoperability between different applications, platforms, and programming languages is a fundamental goal of integration.

Lean Integration: A management system that emphasizes creating value for end customers, continuous improvement, and eliminating waste as a sustainable data and process integration practice.

Measure: A quantitative performance indicator or success factor that can be traced on an ongoing basis to determine successful operation and progress toward objectives and goals.

Metadata: Data about data and data processes. Metadata is important because it aids in clarifying and finding the actual data.

Meta-model: A description of a model. A meta-model refers to the rules that define the structure a model can have. In other words, a meta-model defines the formal structure and elements of a model.

Methodology: A defined, repeatable approach to address a particular type of problem. A methodology typically centers on a defined process but may also include definition of content. May be used interchangeably with the term *method*.

Middleware: Computer software or hardware that connects other software components or systems. This technology evolved to provide for interoperability in support of the move to coherent distributed architectures, which are used most often to support complex distributed applications. It includes Web servers, application servers, and similar tools that support application development and delivery. Middleware is especially integral in managing and optimizing a system-of-systems at the enterprise level. Middleware sits “in the middle” between applications that may be working on different operating systems. It is similar to the middle layer of a three-tier single-system architecture, except that it is stretched across multiple systems or applications. Examples include EAI, ETL, EII, BPM, SOA, and MOM (message-oriented middleware).

Operational data store: A database that is subject-oriented, read-only to end users, current (non-historical), volatile, and integrated; is separate from and derived from one or more systems of record; and supports day-to-day business operations and real-time decision making.

Organization unit: A self-contained unit of resources with line management responsibility, goals, objectives, and measures. Organizations may include external parties and business partner organizations.

Pattern: A common combination of logic, interactions, and behaviors that form a consistent or characteristic arrangement. An important use of patterns is the idea of design templates that are general solutions to integration problems. They will not solve a specific problem, but they provide a sort of architectural outline that may be reused in order to speed up the development process.

Platform: A combination of technology infrastructure products and components on which various application programs can be designed to run.

Process integration: Automation of processes that cut across functional or application boundaries where process state needs to be maintained independently of the underlying application systems or where multiple data consumers or data providers need to be orchestrated as part of a business transaction.

Protocol: The rules governing the syntax, semantics, and synchronization of communication.

Publish/subscribe: A message exchange pattern where a service consumer subscribes to get a notification message from a service provider when a certain condition or state occurs or changes.

Road map: An abstracted plan for business or technology change, typically operating across multiple disciplines over multiple years.

Role: The characteristic and expected behaviors of an individual, derived from his or her responsibilities and preferences in providing value to the organization.

Service: In an organizational context, a set of benefits delivered by a service provider, mostly in close coordination with other service suppliers, commissioned according to the needs of the service consumer, and used by the requesting service consumer for supporting day-to-day business tasks. For example, an ICC provides services to internal consumers that may be other IT or business functions.

Service, information system: A service that is specifically provided by an automated IT-based solution. In an SOA, the IT realization of self-contained business functionality, sometimes also referred to as a Web service. Technically, a service is a description of one or more operations that use messages to exchange data between a provider and a consumer.

Service level agreement (SLA): A formal negotiated agreement between two parties that usually records the common understanding about priorities, responsibilities, and warranties, with the main purpose of agreeing on the quality of the service. For example, an SLA may specify the levels of availability, serviceability, performance, operation, or other attributes of the service (such as billing and even penalties in the case of violations of the SLA).

Service-oriented architecture (SOA): In its most general sense, an approach for architectures where the interfaces are services. In a more specific sense, it is an architectural style for dealing with business processes distributed over a large and heterogeneous landscape of existing and new systems that are under the control of different owners. The key concepts of SOA are services, interoperability, and loose coupling.

System: A set of interacting or interdependent computer hardware and software components forming an integrated unit. In this book, the terms *system* and *application* are often used interchangeably. A business system supports capabilities in a particular business domain (such as finance, marketing, manufacturing, sales, etc.), whereas an integration system supports capabilities in a particular integration discipline (such as data integration, process integration, data quality, business intelligence, etc.). See also the definition for **system-of-systems**.

System of record: The single authoritative, enterprise-designated source of operational data. It is the most current, accurate source of its data.

System-of-systems: The collection of interconnected systems in an enterprise. Modern systems that form systems-of-systems are not monolithic; rather they have five common characteristics: operational independence of the individual systems, managerial independence of the systems, geographical distribution, emergent behavior, and evolutionary development.

Transfer price: The price that one subunit (department or division) charges for a product or service supplied to another subunit of the same organization. Transfer pricing is the mechanism used by ICCs to charge for their services on either a market, cost-plus, or negotiated basis.

Web services: A set of standards that serves as one possible way of realizing an SOA infrastructure. Initially started with the core standards XML, HTTP, WSDL, SOAP, and UDDI, it now contains over 60 standards and

profiles developed and maintained by standardization organizations, such as W3C, OASIS, and WS-I.

Workflow: Similar to a business process; a description of the activities or tasks that have to be done to fulfill a certain business need. Some people differentiate between workflows and business processes by stating that business processes describe more generally what has to be done, whereas workflows describe how activities or tasks should be carried out.