Measuring and Improving Performance

Information Technology Applications in Lean Systems

James William Martin

CRC Press
Taylor & Francis Group

Measuring and Improving Performance

Information Technology Applications in Lean Systems

Measuring and Improving Performance

Information Technology Applications in Lean Systems

James William Martin

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

Productivity Press
Taylor & Francis Group
270 Madison Avenue
New York, NY 10016

Productivity Press is an imprint of Taylor & Francis Group, an Informa business

No claim to original U.S. Government works

Printed in the United States of America on acid-free paper
10 9 8 7 6 5 4 3 2 1

International Standard Book Number: 978-1-4200-8418-4 (Hardback)

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the Productivity Press Web site at
http://www.productivitypress.com

Contents

SECTION 2 PROCESS IMPROVEMENT METHODS

Preface

In an era of globalization, in which supply-chain functions are scattered across the world, measuring operational performance can be difficult. In fact, in the absence of measurement baselines, process improvements are often difficult or impossible to implement. In many situations, an organization may have a great amount of data locked up within its information technology (IT) systems, but it is unable to synthesize or use the data to improve its process workflows. Also, in many organizations there is often little synergy between an organization's IT and operational systems. Our goal must be to make IT systems flexible by integrating them in ways to make them more responsive to external demand and the immediate capacity constraints of an operational system. In other words, IT can be intelligently and even elegantly employed to simplify, standardize, and integrate the components of disparate IT and operational systems to create processes that dynamically respond to changes of customer demand within the limits of internal and external capacity constraints.

Many books have been written to describe methods to develop and deploy IT systems. These include the creation and management of software and hardware systems, as well as their features and functions within the context of measuring and improving operational performance. This has been an evolving process within many organizations over the past several decades. In many applications, IT deployment efforts have been very successful in both manufacturing and service systems. This is a well-known fact that is evident to anyone using a personal computer, the Internet, or one of the many excellent workflow management tools to help manage the flow of information through operational systems. Some of these books have become classics within the IT community. They are also listed in the Suggested Reading sections of each chapter. Examples include Brooks's *The Mythical Man-Month*, DeMarco and Lister's *Peopleware: Productive Projects and Teams*, Schwaber's *Agile Project Management with Scrum*, Hunt and Thomas's *The Pragmatic Programmer: From Journeyman to Master*, and Fowler's *Refactoring: Improving the Design of Existing Code*. The authors provide practical advice for creating software, and managing its development and commercialization activities. A common goal is to reduce the development cycle time of software projects to improve the overall effectiveness and

efficiency of managing, creating, deploying, and maintaining software code. In this context, effectiveness is defined as doing the right work, and efficiency is doing the right work the right way.

However, from another perspective, the basic concepts, tools, and methods of Lean systems are also relevant to improving process effectiveness and efficiency. A major focus of this book, therefore, is to discuss the synergy gained from using best-in-class software management design and project management methods within Lean systems. In Chapter 5 we will learn some basic Lean concepts and then show how they can be used within systems that are heavily dependent on IT. From a Lean perspective, some relevant sources of information include Goldratt and Cox's *The Goal: A Process of Ongoing Improvement*, Suri's *Quick Response Manufacturing: A Companywide Approach to Reducing Lead Times,* and Womack and Jones's *Lean Thinking: Banish Waste and Create Wealth in Your Organization*. These books are also listed in the Suggested Reading section of Chapter 5. Our approach will be to link these diverse initiatives and concepts using three perspectives: project management, the efficient development of software code, and the application of both to improve the effectiveness and efficiency of Lean systems using IT. The project management perspective will use the concepts, tools, and methods of Agile Project Management (APM) with scrum. Some key titles include Augustine's *Managing Agile Projects* and Schwaber's *Agile Project Management with Scrum*. The perspective related to the efficient design of software and how it relates to Lean concepts will be discussed within the context of Hunt and Thomas's *The Pragmatic Programmer*, Fowler's *Refactoring,* and similar sources. Finally, the third key perspective, improving the effectiveness and efficiency of Lean systems, will be discussed based on several key authors, including those mentioned earlier. Many of these books are listed as suggested reading in more than one chapter.

The goal is to discuss these perspectives and show how they can be used synergistically to integrate IT applications within the operational systems of Lean organizations. The discussion will focus on two major areas. First, IT applications should be deployed within organizations to increase their flexibility in responding to external demand within the constraints of available resources. In other words, IT applications should be deployed effectively and efficiently to help achieve the goals of a Lean enterprise. Agile Project Management with scrum and related project management methods will be integral to these discussions. Second, to the extent that IT systems currently exist within an organization, it may be possible to modify them to accelerate the deployment of Lean systems to improve operational performance. Modifications to current IT systems are often necessary to make them more useful in a Lean work environment. This is because they represent a heavy investment in infrastructure and cannot be easily replaced due to high capital costs, or they may be necessary for practical reasons. An example of the latter situation is the use of a manufacturing resource planning II (MRPII) system to manage and control the manufacturing of thousands of products. An MRPII system or module converts independent demand from a master production schedule (MPS) using

bill of material (BOM), inventory file, and related information to calculate the quantities of dependent-demand items. These dependent demand items are then ordered from suppliers or manufactured by an organization's production facilities. However, an MRPII module can be decoupled at lower levels to make it more responsive to changes in external demand and available resources to mimic, to some degree, a pull production system at a work-cell level. At higher levels of a BOM, however, the MRPII module can still coordinate production activities throughout a supply chain. It is through this type of out-of-the-box thinking that leading-edge concepts can help to integrate IT applications to facilitate the deployment of Lean systems. I wrote this book because I believe it will be useful to show how IT can support Lean applications in both manufacturing and service systems to increase their productivity.

I thank my editor, the late Raymond O'Connell, for encouraging me to publish this book, as well as the reviewers of the book who made constructive comments that changed its direction to a more practical and hands-on approach. I also thank my son, Paul, a computer science student attending Johns Hopkins University (class of 2011) for reading several key chapters in this book for their technical content. I also thank my daughter, Krysta, a student at Emory University School of Law and my wife, Marsha, for their support as well as several of my professional friends. These people include Peri Kaae, Manager of Process Improvement, Florence Woo, Linc Markham, and William Johnson. I also want to thank Elaine Kowansky and Ram Josyula, formerly of General Electric and who are now owners of Gelrad Consulting. Finally, I thank my graduate students and clients, who have provided the inspiration for this book.

Introduction

> Once we rid ourselves of traditional thinking we can get on with creating the future.
>
> **—James Bertrand**

This book has been written for Lean and Six Sigma practitioners to help them understand some of the important concepts related to the creation or modification of software to support process management and improvement activities in Lean systems. In other words, it is not a reference for programmers but is intended instead as a source of information for Lean practitioners who have been assigned the responsibility of creating a new information technology (IT) application; improving the performance of an existing IT application; or integrating an IT application within an operational system to improve its performance relative to time, quality, or cost.

Many operational systems are now global and globalization has been evolving over the past several decades. This evolutionary process has been described by several authors and particularly in the books *Why Globalization Works* by Martin Wolf and *Adam Smith in Beijing: Lineages of the 21st Century* by Giovanni Arrighi. These authors describe the impact of globalization on business organizations as well as societies in general. In support of these globalization concepts and framework, I write from an operational perspective with a focus on increasing organizational productivity through the application of IT to improve Lean production systems. In the synthesis of the critical ideas put forth by these and other authors, my goal has been to develop a conceptual framework from which to move Lean systems to their next level across the various countries and cultures who are participants of global supply chains, which often consist of combinations of service and manufacturing processes. In other words, as the world has globalized and evolved into service economies, the need to develop and deploy IT to increase operational efficiency and, in turn, organizational productivity has become more critical. These topics are especially important in a world that has come to rely on increasingly higher levels of productivity to maintain its standard of living in an environment of increasing resource constraints.

However, although an organization's goal is to improve its overall operational efficiency, improvement activities must begin at lower levels to ensure that operational systems are designed to efficiently respond to changing customer requirements as well as customers' evolving value expectations. This work must be able to be done anywhere in the world. Therefore the operational systems that are supported by IT must also be designed to simultaneously service customers of many different countries, languages, and cultures. For example, every day and in a routine manner, service workers in the United States, India, China, and many other locations around the world interact through common workflow systems. These workflow systems consist of many diverse functions, and may also support several different languages in diverse cultural and political environments. For example, workflow systems routinely manage the operational systems that create and manufacture products and services. The ordering and shipment of computers, consumer electronics, and books are typical examples of global workflows that touch many diverse customers. Global workflows also help manage the internal day-to-day operations of numerous organizations. Examples include managing accounts payables, receivables, and similar financial functions of an organization.

In response to these types of organizational needs, diverse IT applications have been successfully deployed across global supply chains using a variety of project management methodologies. As an example, if a deployment's goal was to create a major software and hardware product, then classical project management methods such as the IBM® Rational Unified Process (RUP)®, which are designed to manage large and complex projects, might have been successfully used by an organization. However, if modifications to a current IT system were required or smaller IT project applications needed to be managed, then perhaps Agile Project Management (APM) with scrum methods were used to ensure that a project met its schedule, cost, and performance targets. Although management of IT projects can be complicated, overly complex project management methods need to be avoided in favor of choosing the best set of management methods to ensure a project's success. In this context, several excellent sources of information include *Peopleware: Productive Projects and Teams* by DeMarco and Lister and *The Mythical Man-Month* by Brooks. These classical references describe key elements of IT project management, including common fallacies related to team formation and resource management. They also show how to improve a project's effectiveness and efficiency using proven tools and methods. A major goal of this book is to improve IT management relative to its impact on Lean systems. Related goals are the efficient design and deployment of software as well as the application of these concepts to improve a Lean system. We will integrate these concepts with those discussed in other sources such as the suggested readings at the end of every chapter. A key vehicle to achieve the goal of effective project management will be an in-depth discussion of APM and scrum in the context of Lean IT project management, as well as Lean concepts and applications. *Managing Agile Projects* by Augustine and *Agile Project Management with Scrum* by Schwaber are excellent references that describe APM and scrum. Software

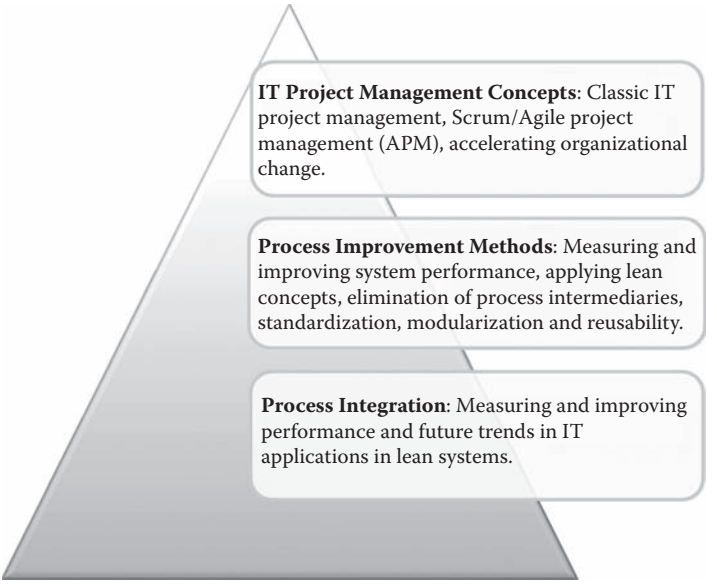

Figure I.1 Focus of the book.

design principles will also be discussed in the context of their Lean analogues. The discussion of Lean applications will build on these key concepts.

This book is divided into three major parts, as shown in Figure I.1, consisting of eight chapters. The first part consists of two chapters that discuss classic IT and APM project management methods. This information is important because IT applications in Lean systems can easily go off track, resulting in a longer project implementation cycle time and higher costs. We will discuss two project management methods that have been shown to be very successful in practice. These will include elements of classical project management as well as RUP, and then APM with scrum. Understanding the strengths of these approaches can make project management more effective. As an example, large-scale projects that create and deploy complex hardware and software systems usually require a higher degree of project structure because the team participants are numerous and are often scattered across geographically dispersed work locations. Also, there are many hardware and software subsystems that must be created, integrated, tested, and deployed in an integrated manner to ensure they will interface and function correctly in a final product. In contrast, the creation and deployment of small- to moderate-size IT system applications, or modifications to current systems, are usually best implemented using APM with scrum. APM methods require that a system's functionality be produced at frequent intervals using activity backlogs and scrum sprints. The expectation of customers and key stakeholders is that over weeks or months their requirements will be met. In some IT applications a combination of both project

management methodologies may also be useful. In summary, APM with scrum can be used at a project work activity level either as a standalone project management strategy or within an RUP project management structure.

In the second part of this book, tools and methods useful for process improvement are discussed in Chapters 3, 4, 5, and 6. The topics include accelerating organizational change, translating customer requirements, applying Lean concepts, and creating and deploying software to support Lean systems. The major goal of these chapters is to move from the discussion of project management to an important discussion associated with the capture of customer needs and requirements, and to translate these requirements into specifications. The discussion of Lean topics in Chapter 5 is necessary to provide context for the various IT topics that are being discussed to show the similarities of each approach. The information contained within Chapter 6 is designed to provide Lean practitioners with some understanding of key topics important to the creation of software code. This information may be useful either when Lean practitioners assist IT teams or if they require the assistance of IT professionals to implement Lean systems within their process workflows.

Chapters 7 and 8 complete the third part of this book. Chapter 7 contains examples from manufacturing and service industries that integrate classical Lean tools and methods with various IT applications. Chapter 8 discusses the measurement and improvement of Lean systems within a Lean environment. The unifying concept of Chapter 8 is process integration. Process integration is discussed relative to sharing information using performance dashboards and the global integration of IT systems in support of operations.

Project Management Methods

Table I.1 describes several project management methods used to manage the creation, modification, and deployment of software. Several of these project management methods use, with some modifications, common tools and methods. Most people are familiar with classical project management, which is a set of formal tools and methods used to plan and manage resources to achieve goals and objectives within schedule and budget. These tools and methods include project charters, activity scheduling charts (Gantt charts), issue lists, and similar methods to capture customer requirements and manage project resources to achieve a project's schedule, cost, and performance targets. Additional tools and methods include work breakdown structure (WBS) charts as well as stakeholder and risk analysis tools. There are many other tools, methods, and templates used to manage projects in a classical sense, and several of these will be discussed in Chapter 1.

The Waterfall Model is the second project management method listed in Table I.1. This model has fallen into general disuse because it employs a sequential software development process in which client requirements and their translation

Table I.1 IT Project Management Methods

Method	*Major Characteristics*
Classical project management	A set of formal tools and methods used to plan and manage resources to achieve goals and objectives within schedule and budget. Typical tools and methods include project charters, Gantt charts, a work breakdown structure (WBS), and stakeholder and risk analyses.
Waterfall Model	A sequential software development process in which client requirements and their translation into software code is developed in sequence using seven phases: obtaining client requirements, software code design, construction, integration of software modules, testing and error correction, client installation, and subsequent maintenance of the software product. All activities are completed within their phase prior to moving to the next phase.
IBM® Rational Unified Process (RUP)	A software development process that emphasizes development of code in iterations, which balance stakeholder requirements through high collaboration across teams. The process is divided into four phases: inception, elaboration, construction, and transition. Major methods include requirements analysis, building and modeling software code through implementation, and testing. Related project management methods include managing risk and resources according to the project's schedule and cost targets. Key tools include an assessment at each software development milestone (iteration), project status metrics, a list of issues and problems, work orders to delegate work task activities, and periodic project reviews.
Lean project management	A set of formal tools and methods that translate customer value elements into a process to identify non-value-adding work tasks to simplify, standardize, and mistake-proof a process. The emphasis is on hands-on and highly visible data collection and analysis.
Agile Project Management (APM)	A set of methods that emphasize the rapid delivery of working software code to clients using small collocated and self-organizing teams that have daily contact with their clients and other key stakeholders.

continued

Table I.1 (continued) IT Project Management Methods

Method	*Major Characteristics*
Agile Unified Process (AUP)	Uses agile project methods in the context of the Rational Unified Process with an emphasis on the use of a development release for quality assurance testing and a subsequent production release. Documentation that is not integrated within software code is kept simple and only used as necessary.
Scrum	A project management approach to running software projects using practices and predefined roles and responsibilities to conduct "sprints" having a duration of 15 to 20 days to complete useable software code. Activities include highly interactive client and other stakeholder involvement. Typical methods include scrum meetings, activity backlogs, and sprint backlogs.
Extreme Programming (EP)	A set of methods that attempt to create an initial solution to client requirements that is simple and testable. In fact, requirements are written directly in the form of automated tests, which must be passed at each integration of code development. Code is written and tested by the client and team, and additional product functionality is added only when necessary using an incremental approach. Typical methods include pair programming, test-driven code development, incremental releases of code, and adherence to standard coding standards.

into specifications and software code is sequentially developed using seven phases. These phases are obtaining client requirements, the design of software code, its construction, the integration of software algorithms and modules, software testing and error correction, client installation, and subsequent maintenance of the software product. Since all project activities must be completed within their current phase prior to moving to the next phase, project cycle times tend to be longer than if APM with scrum or other methods were used as a project management framework. Another way to think of the Waterfall methodology is that completed work activities are thrown "over the fence" to the next project team. In contrast, APM with scrum and several other project management methods facilitate working concurrently across teams to improve a project's coordination and communication to enable a higher degree of project management and control. Concurrent implies that project teams communicate in parallel to provide clear visibility of a project's status relative to its ongoing or completed work activities. A Waterfall project management approach also increases a project's cycle time and costs and results in

lower quality levels, as opposed to using a concurrent approach that engages team members, customers, and suppliers on a frequent basis.

IBM developed the RUP to deploy complicated hardware and software systems as one answer to the many issues surrounding classic project management and Waterfall methods. RUP is designed for the project management of new hardware and software systems that create new features and functions. In this context, new user requirements are translated into hardware and software systems that span numerous project teams. In other words, RUP was created to support the creation of new and complex IT architecture. RUP is a software development process using a project management approach that emphasizes the development of software code in discrete iterations. This approach attempts to balance stakeholder requirements by promoting a high degree of collaboration across teams. Key stakeholders, including various organizational functions, suppliers, and customers, have an interest in a project's ongoing work activities, resource requirements, and goal achievement. Key characteristics of the RUP project management approach include the continuous adaptation of project management methodologies in response to customer needs, the balancing of key stakeholder priorities, a focus on the quality of written software code, team communication, concurrent collaboration, and proving that customer requirements (regarding new and useful features and algorithm functions) are met on a frequent basis. This is in contrast to waiting until close to the end of a project to conduct full system testing. Another useful characteristic of RUP methodology is its focus on elevating the abstraction level of software coding. It should also be noted that this is a common best-in-class practice of modern software design. An elevation of a system's abstraction level will be discussed within the context of APM and scrum in Chapter 6, because this is one of many ways to improve the efficiency of creating standardized and flexible software code.

RUP is divided into four phases: product inception, elaboration, construction, and transition. In the inception phase of a project, the expected resources and their costs, which are based on estimated work activities, are compared to budgeted amounts to ensure that a project, as currently defined, will meet its financial objectives. If a project is over budget, then redesign of a product and its work activities may be necessary, or the project's budget may need to be increased prior to proceeding to its next phase. In the elaboration phase, a product's basic concept and requirements are created and developed as defined by its features and functions. This detail includes customer and stakeholder requirements, use cases, and examples of software architecture and prototypes, which show how the required product features and functions will be achieved within the constraints of a project's resources and with an acceptable level of risk. Prior to leaving the elaboration phase, a business plan for a project should be revised based on available information regarding product testing and other performance requirements. In the construction phase of a project, software code is created and a product's features and functions are fully developed and tested. If a project is large and complex, then subteams will be testing their portions of the software code in a concurrent manner and then sharing testing