

Lean Software Development in Action

Andrea Janes • Giancarlo Succi

Lean Software Development in Action

 Springer

Andrea Janes
Giancarlo Succi
Libera Università di Bolzano
Bolzano
Italy

ISBN 978-3-662-44178-7 ISBN 978-3-642-00503-9 (eBook)

DOI 10.1007/978-3-642-00503-9

Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014953961

© Springer-Verlag Berlin Heidelberg 2014

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

We wrote this book with the idea to show a practical implementation of Lean software development, gluing together well-proven tools to provide a way to develop Lean. The message this book wants to convey is the utilization of goal-oriented, automated measurement for the creation of a Lean organization and the facilitation of Lean software development.

Since its conception in the mid-1950s, Lean thinking has been very successful in manufacturing: it has helped organizations to focus on value-providing activities, to identify unnecessary ones, and therefore to increase the efficiency and effectiveness of the overall process. There are several proposals on how to translate Lean principles into software engineering practices, starting from the Agile Manifesto and the pioneering work of Kent Beck (see Chap. 4), of Mary and Tom Poppendieck (see Chap. 6), and of many others.

It is a fact that to be successful, a Lean orientation of software development has to go hand in hand with the business strategy of the company as a whole. To achieve such a goal, there are two interrelated aspects that require special attention: measurement and experience management.

Measurement means to describe real-world processes according to clearly defined rules to understand, control, and improve (see Chap. 9). Now, “Lean software company thinking” requires a comprehensive measurement program: measurement is a cornerstone of Lean thinking, as well evidenced by the founders of the Lean approach, such as Taiichi Ōno, James P. Womack, and Daniel T. Jones (see Chap. 1).

Managers of software companies have often perceived measurement as a waste of time, mostly because the measurement process required a lot of effort from developers and managers, and often the results of a measurement program could not be used to steer effectively the direction of the business. In the last decade, a new approach to software measurement has emerged, where, on one side, it has become evident that a successful software measurement program ought to be introduced in a company-wide measurement approach to promote process improvement, as discussed by the CMMI (see Chap. 3) and several other recent software process

improvement initiatives, and, on the other side, the measurement process has become much less invasive and therefore effort consuming (see Chap. 9).

Measurement is a necessary step in systematic quality approaches like Six Sigma or in performance management instruments like the Balanced Scorecard (see Chap. 3). A methodical approach to measurement as proposed in this book will allow to introduce quality and performance management instruments that rely on it.

The second aspect requiring special attention is experience, i.e., valuable, stored, specific knowledge that was acquired in a previous problem-solving situation (see Chap. 8).

Software development organizations have always been concerned with collecting and institutionalizing the experience of its developers and engineers, so that it would not be too dependent on its key people and the turnover would not affect it too much. The experience of these years have shown that Lean software organizations are even more dependent on the skills of individual people.

It appears therefore critical to promote the institutionalization of experience across the entire company. The studies that the authors have done on the field have evidenced that the collected software measures can be extremely instrumental to achieve such a goal.

Altogether, in this book the authors provide the necessary knowledge to establish “Lean software company thinking” taking advantage of the most recent approaches to software measurement.

The main target group of the book consists of people working in the field of software engineering that want to understand how to obtain an efficient and effective software development process. This group includes developers, managers, and students following an M.Sc. curriculum in software engineering.

This book particularly applies to small and medium enterprises—the dominating organizational form in the European software sector—that do not have the resources to put in place the mentioned processes of collecting and institutionalizing the experience of its developers and engineers. This creates a high dependency on its key people, and turnover can severely challenge the ability to continue to provide a given product or service to the market. Additionally, not institutionalizing the own experiences means wasting an important opportunity for small and medium enterprises to maintain or improve their position in the market.

A comprehensive, company-wide measurement approach is exactly what (even very small) companies need to align the performed activities to the needs of the stakeholders, to the business strategy, etc. With the automatic, non-invasive measurement proposed in this book—this is what we mean by “Lean Software Development in action”—more companies will be able to optimize their software development process towards Lean.

The discussion on how to transfer Lean concepts into software development is still ongoing. To support this process, we aim to inspire others to follow our line of research to apply measurement to understand the effects of introduced changes in the software development process even at an early stage. Measurement helps

to understand and to reason about the obtained desired and undesired changes. Moreover, it points out opportunities to improve.

We present our approach to Lean Software Development in three parts:

1. The first part illustrates what the term “Lean Production” means, why we think it is useful to transfer Lean concepts into software engineering, and which approaches exist to transfer the Lean concept into software engineering.
2. The second part illustrates the tools we use to achieve Lean Software Development: Non-invasive Measurement, the Goal Question Metric approach, and the Experience Factory.
3. The third part illustrates how we combined the different tools to enable Lean Thinking in software development.

Background Knowledge

Whenever we thought that some background knowledge is interesting or useful in a particular part of the text, we added it in a box like this.

A last but important note: throughout this book, wherever the masculine form is used, it applies to the feminine form as well.

Bozen, Italy
May 2014

Andrea Janes
Giancarlo Succi

Acronyms

ABC	Activity Based Costing
AJAX	Asynchronous JavaScript and XML
ANSI	American National Standards Institute
API	Application Programming Interface
PMI	Project Management Institute
AT&T	American Telephone and Telegraph Company
BC	Before Christ
CC	Cyclomatic Complexity
CD	Compact Disk
CIO	Chief Information Officer
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
DAD	Disciplined Agile Delivery
DMAIC	Define, Measure, Analyze, Improve, Control
DSDM	Dynamic Systems Development Method
ETL	Extract Transform Load
FSS	Flagship Software Services, a company in our fictional story
GQM	Goal Question Metrics
GUI	Graphical User Interface
HTML	HyperText Markup Language
IBM	International Business Machines
ICSE	International Conference on Software Engineering
ISO	International Organization for Standardization
IT	Information Technology
IQ	Intelligence Quotient
JSP	Java Server Pages
LOC	Lines Of Code
NASA	National Aeronautics and Space Administration
NPV	Net Present Value
PC	Process Costing
PDCA	Plan Do Check Act

PDSA	Plan Do Study Act
PSP	Personal Software Process
QIP	Quality in Practice
RCA	Resource Consumption Accounting
REST	REpresentational State Transfer
RFID	Radio-frequency identification
RAD	Rapid Application Development
ROI	Return on Investment
SOA	Service Oriented Architecture
SWOT	Strengths, Weaknesses, Opportunities, and Threats
UML	Unified Modeling Language
VP	Vice President
WMC	Weighted Methods per Class
WWI	World War I
WWII	World War II
XP	Extreme Programming
XXX	The boss in our fictional story

Contents

Part I Motivation for Lean Software Development

1	Introduction	3
1.1	Introduction.....	4
1.2	Tame and Wicked Problems	8
1.3	Software Development Is a Wicked Problem	10
1.4	Taylorism and Software Development	11
1.5	Summary.....	15
	Problems	16
	References.....	16
2	The Lean Revolution	19
2.1	Introduction.....	20
2.2	Henry Ford.....	23
2.3	Taiichi Ōno and the Toyota Production System.....	26
2.4	Creating a “Radiography” of the Production Process	27
2.5	Worker Involvement.....	31
2.6	“Pull” and Not “Push”	33
2.7	The Right Parts at the Right Moment at the Right Place	35
2.8	The Right Information at the Right Moment at the Right Place ...	39
2.9	Quality Management	41
2.10	Summary.....	45
	Problems	46
	References.....	47
3	Towards Lean Thinking in Software Engineering	49
3.1	Introduction.....	51
3.2	Value	52
3.2.1	Risk as a Value-Maximizing Strategy	54
3.3	Knowledge.....	57
3.4	Improvement.....	59

3.5	“Push” vs. “Pull” in Software Engineering: “Requirements-First” Development	62
3.6	“Push” vs. “Pull” in Software Engineering: “Bottom-Up” Development	64
3.7	Summary	65
	Problems	66
	References	66
4	Agile Methods	69
4.1	Introduction	72
4.2	Keeping the Process Under Control	76
4.3	Job Enrichment	79
4.4	Endogenous and Exogenous Control Mechanisms	81
4.5	Synchronizing the Flow of Work of Multiple People	82
4.6	Extreme Programming (XP): A Paradigmatic Example of Agile Methods	83
4.7	The Building Blocks of XP	84
4.8	The XP Practices	87
4.8.1	Business Practices	89
4.8.2	Integration Practices	89
4.8.3	Planning Practices	89
4.8.4	Programming Practices	91
4.8.5	Team Practices	92
4.8.6	Uncategorized, Generic Practices	93
4.9	Control and Coordination Mechanisms	95
4.10	Summary	99
	Problems	99
	References	100
5	Issues in Agile Methods	103
5.1	Introduction or “the Hype of Agile”	105
5.2	The Dark Side of Agile	110
5.3	The Skepticism Towards Agile Methods	116
5.4	The Zen of Agile	120
5.5	Summary or “What Stops us from Moving from Agile Towards Lean Software Engineering?”	125
	Problems	126
	References	126
6	Enabling Lean Software Development	129
6.1	Introduction	130
6.2	Existing Proposals to Create “Lean Software Development”	130
6.3	Share a Common Vision	134
6.4	Deprive Gurus of Their Power	142
6.5	Disarm Extremists	146
6.6	Summary	146

Problems	147
References	147

Part II The Pillars of Lean Software Development

7 The GQM⁺ Strategies Approach	151
7.1 Introduction	152
7.2 What Can We Measure?	154
7.3 What Should We Measure?	155
7.4 Applying the GQM Step-By-Step	160
7.5 Alignment	164
7.6 Summary	168
Problems	169
References	169
8 The Experience Factory	171
8.1 Introduction	172
8.2 Why Plan-Do-Study-Act Does Not Work in Software Engineering	172
8.3 The Experience Factory	174
8.3.1 Work Distribution	175
8.4 The QIP Step-by-Step	177
8.5 The Role of Measurement	181
8.6 Summary	183
Problems	183
References	184
9 Non-invasive Measurement	187
9.1 Introduction	188
9.2 Does Measurements Collection Pay Off?	195
9.3 Non-Invasive Measurement	197
9.4 Implementing Non-invasive Measurement	207
9.5 The “Big-Brother” Effect of Non-invasive Measurement	211
9.6 Summary	214
Problems	214
References	215

Part III Lean Software Development in Action

10 The Integrated Approach	221
10.1 Introduction	222
10.2 The Role of Autonomation	228
10.3 Closing the Loop with an Andon Board for Lean Software Development	237
10.3.1 Visualizing the “Right” Data	239
10.3.2 Visualizing Data “Right”	240
10.3.3 Putting the Pieces Together	243

10.4	Summary.....	245
	Problems	246
	References.....	246
11	Lean Software Development in Action	249
11.1	Introduction.....	249
11.2	Evaluating Action Research	254
11.3	Introducing Measurement Programs in Companies	256
11.3.1	Plan.....	257
11.3.2	Act.....	258
11.3.3	Observe	261
11.3.4	Reflect	263
11.3.5	Revise Plan.....	264
11.4	Case 1: Exploration or Exploitation?.....	264
11.4.1	Theoretical Framework	266
11.4.2	The Study	270
11.4.3	Results	277
11.4.4	Discussion	278
11.5	Case 2: Non-invasive Cost Accounting	279
11.5.1	Theoretical Framework.....	289
11.5.2	The Study	291
11.5.3	The Role of the Experience Factory in Cost Accounting	317
11.5.4	Results	318
11.5.5	Discussion	318
11.6	Case 3: Developing a Lean GQM Graph.....	319
11.6.1	Theoretical Framework.....	319
11.6.2	The Study	321
11.6.3	Results	348
11.6.4	Discussion	348
11.7	Summary.....	349
	Problems	349
	References.....	350
12	Conclusion.....	355
12.1	Introduction.....	355
12.1.1	Lessons Learned	356
	Problems	357
	References.....	357
A	If Architects Had to Work Like Software Developers	359
B	A Possible Architecture for a Measurement Framework	361
B.1	Scenarios.....	362
B.2	Logical View	364
B.3	Physical View.....	372

B.4 Process View 373

 B.5 Development View 373

Solutions..... 375

 References..... 389

Index..... 391