"If you want to understand how to lead a Continuous Delivery or DevOps transformation in your company, there's no better book than this. Concise, practical, and based on hard-won executive experience, this book is essential reading for every IT executive."
—**Jez Humble**, VP, Chef

Applying Agile and DevOps Principles at Scale

LEADING THE TRANSFORMATION

PDF COMPANION TO THE AUDIO BOOK

GARY GRUVER *and* TOMMY MOUSER

foreword by GENE KIM

LEADING THE TRANSFORMATION

Applying Agile and DevOps Principles at Scale

Gary Gruver and Tommy Mouser

Foreword by Gene Kim

IT Revolution
Portland, OR

PDF COMPANION TO THE AUDIO BOOK

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

Third Edition
Printed in the United States of America

23 22 21 20 19   3 4 5 6 7 8 9 10

Cover design: Devon Smith
Book design and illustrations: Brian David Smith and Mammoth Collective

ISBN: 978-1-942788-01-0
eBook ISBN: 978-1942788-02-7
Kindle ISBN: 978-1942788-03-4

Publisher’s note to readers: Although the authors and publisher have made every effort to ensure that the information in this book is correct, the authors and publisher do not assume and hereby disclaim any liability to any party for any loss, damage, or disruption caused by errors or omissions, whether such errors or omissions result from negligence, accident, or any other cause.

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

LEADING THE TRANSFORMATION

CONTENTS

Figures

Further Reading

About the Authors

Acknowledgments

LIST OF FIGURES

Figure 1 Enterprise-level Transformation

Figure 2 Waterfall Development Model

Figure 3 Agile Development Model

Figure 4a Cycle Time and Cost-Drivers 2008

Figure 4b Cycle-Time Driver Improvements

Figure 4c Development Cost Driver Improvements

Figure 4d State-of-the-Art Firmware Development Model

Figure 5 MM30 Checkpoint Status

Figure 6 Enterprise-Level Continuous Improvement

Figure 7 Long-Term Predictability for Software Schedules

Figure 8 Leftover Capacity

Figure 9 Futuresmart Firmware User Stories Per Sprint—Estimating Short-Term Features Based on Delivery

Figure 10 Train Wreck Blocking Code Progession

Figure 11 Gating Commits

Figure 12 Ensuring the Architecture/Build System is Ready

Figure 13 Continuous Delivery—Picking the Right Tool for the Job

Figure 14 Scripted Environment Architecture

Figure 15a Finding the Offending Component Quickly Improves Productivity

Figure 15b Finding the Offending Component Quickly Improves Productivity

Figure 16 Building Up the Enterprise System

Figure 17 Building Up the Enterprise System

Figure 18 Application Gating

Figure 19 Loosely Coupled Architecture Deployment Pipeline

Figure 20 Release Readiness

Figure 21 Farming the Build Acceptance

Figure 22 Continuous Integration and Test System

Figure 23 Embedded System Pipeline

Figure 24 Enterprise-level Transformation

FIGURE 1: ENTERPRISE-LEVEL TRANSFORMATION

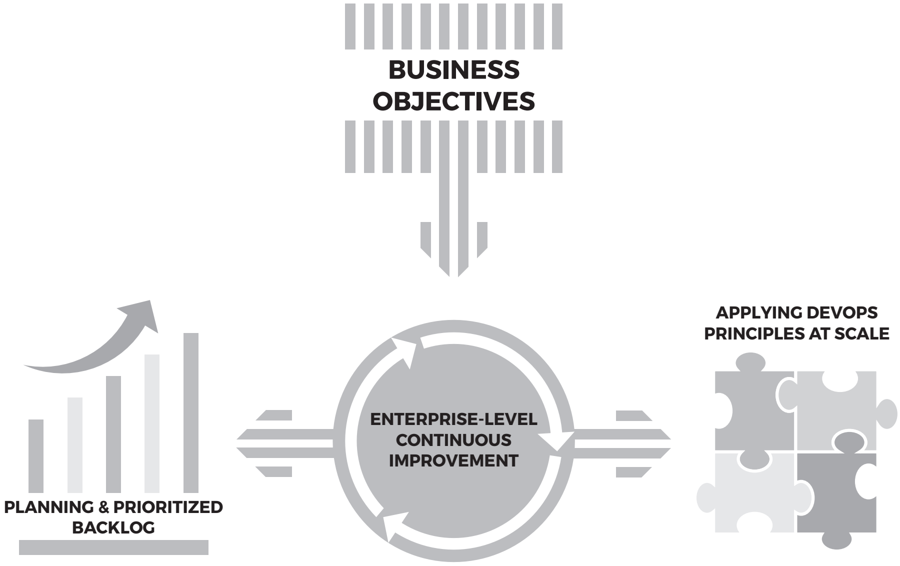
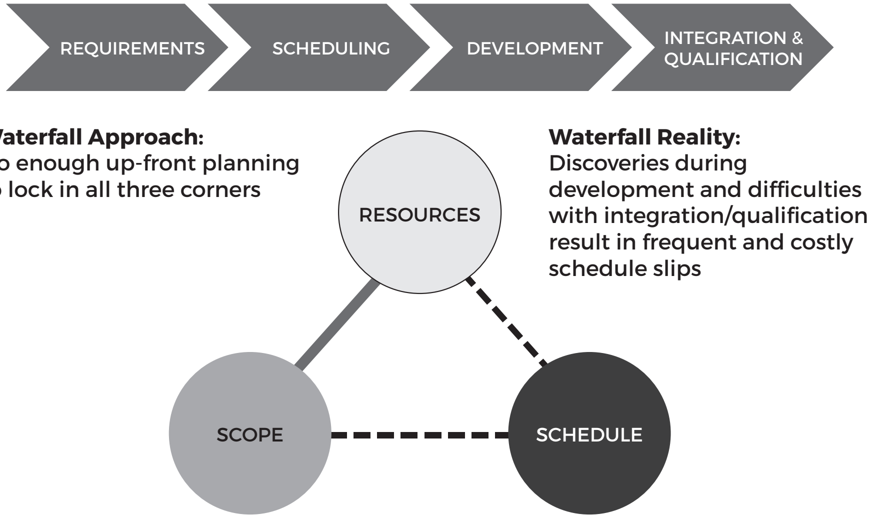

BUSINESS OBJECTIVES
PLANNING & PRIORITIZED BACKLOG
ENTERPRISE-LEVEL CONTINUOUS IMPROVEMENT
APPLYING DEVOPS PRINCIPLES AT SCALE

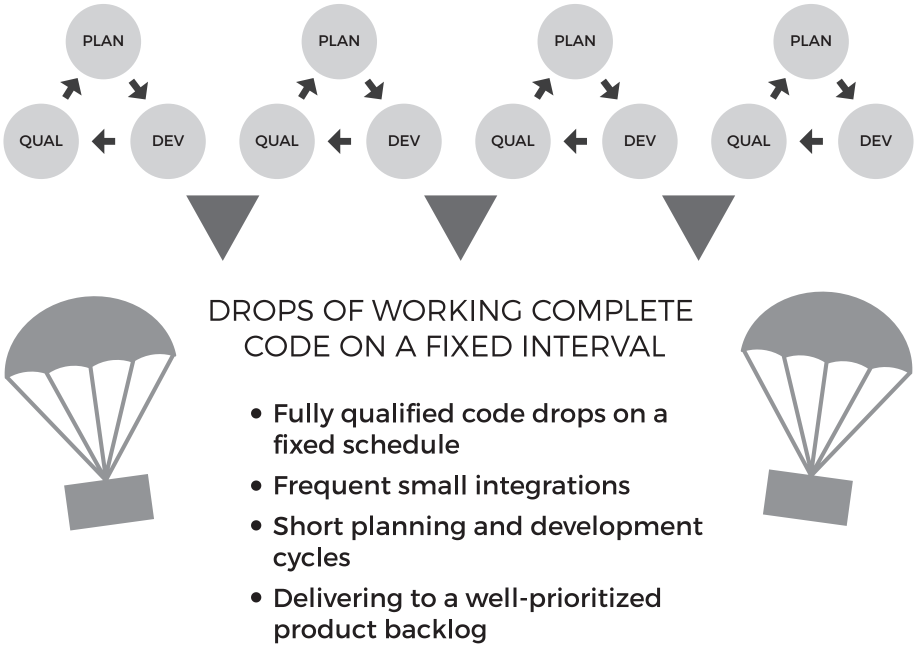
FIGURE 2: WATERFALL DEVELOPMENT MODEL



Waterfall Approach:
Do enough up-front planning to lock in all three corners

Waterfall Reality:
Discoveries during development and difficulties with integration/qualification result in frequent and costly schedule slips

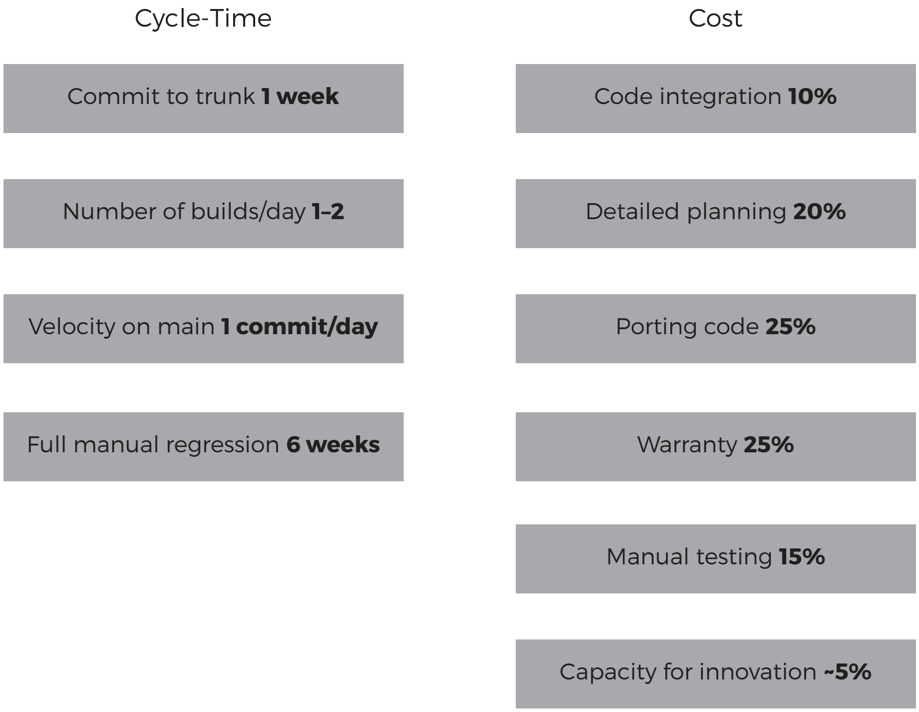
FIGURE 3: AGILE DEVELOPMENT MODEL



DROPS OF WORKING COMPLETE CODE ON A FIXED INTERVAL

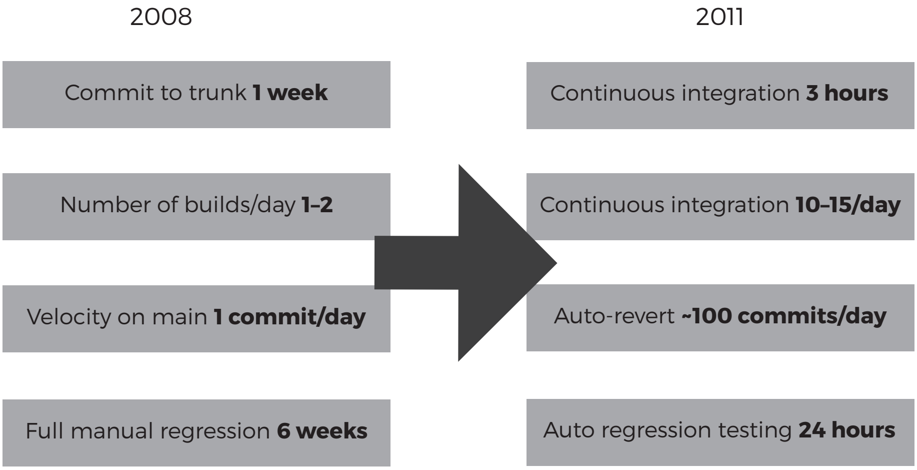
- Fully qualified code drops on a fixed schedule
- Frequent small integrations
- Short planning and development cycles
- Delivering to a well-prioritized product backlog

FIGURE 4A: CYCLE-TIME AND COST DRIVERS 2008



Cycle-Time	Cost
Commit to trunk **1 week**	Code integration **10%**
Number of builds/day **1–2**	Detailed planning **20%**
Velocity on main **1 commit/day**	Porting code **25%**
Full manual regression **6 weeks**	Warranty **25%**
	Manual testing **15%**
	Capacity for innovation **~5%**

FIGURE 4B: CYCLE-TIME DRIVER IMPROVEMENTS



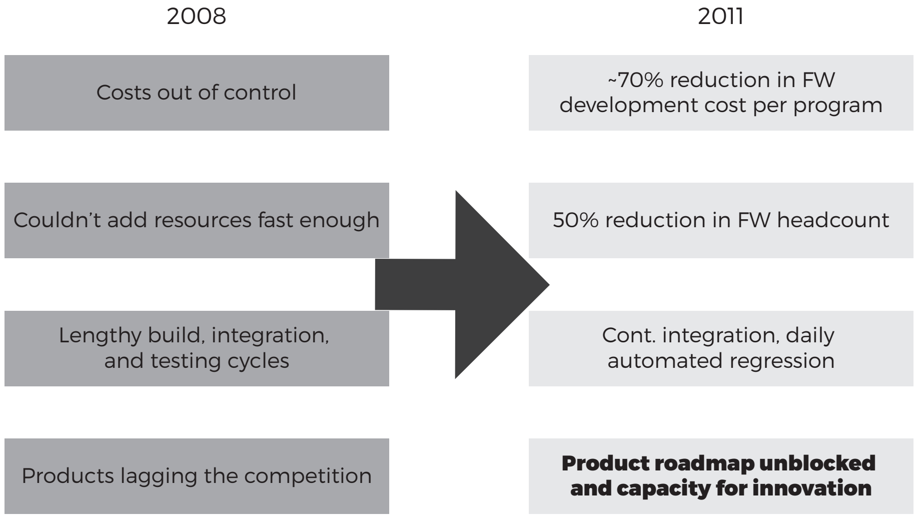
2008	2011
Commit to trunk **1 week**	Continuous integration **3 hours**
Number of builds/day **1–2**	Continuous integration **10–15/day**
Velocity on main **1 commit/day**	Auto-revert **~100 commits/day**
Full manual regression **6 weeks**	Auto regression testing **24 hours**

FIGURE 4C: DEVELOPMENT COST DRIVER IMPROVEMENTS

2008	2011
Code integration **10%**	Continuous integration **2%**
Detailed planning **20%**	Agile planning **5%**
Porting code **25%**	One main branch **15%**
Warranty **25%**	Warranty **5%**
Manual testing **15%**	Most testing automated **5%**
Capacity for innovation **~5%**	**Capacity for innovation ~40%**

2011 intentionally not = 100%. The difference was used for further process improvements.

FIGURE 4D: STATE-OF-THE-ART FIRMWARE DEVELOPMENT MODEL



2008	2011
Costs out of control	~70% reduction in FW development cost per program
Couldn’t add resources fast enough	50% reduction in FW headcount
Lengthy build, integration, and testing cycles	Cont. integration, daily automated regression
Products lagging the competition	**Product roadmap unblocked and capacity for innovation**

FIGURE 5: MM30 CHECKPOINT STATUS

RANK	THEME	EXIT CRITERIA
0	Quality	- P1 open < 1wk - CAT 100% pass - L2 24hr response
1	1st bit release on new Arch for WinXP scanner	**A) Final P1 defects fixed** 2 remaining. B) Duration error rate per 10K: 0.3 (sim), 0.35 (emul), **0.4 (product)**
2	Ensuring common code stability on WinCE & the CE products supported	A) Customer accpt level (CAT) tests 100% passing on CE B) Test coverage appropriate for CE added to L1 (Terrese) C) All L2 pillars 98% pass – w/ coverage for high-value Product turn-on reqts for the CE products D) L4emu test pillars – LLFW (Arch), copy/PDL (Brian), PD (Hugh) E) Garnet L3 CAT in place with at least L4 CAT equivalence
3	Supporting the product reqs for the MIPS based products on CE	A) Calibration dependencies (Kimberly/Brian/Ted/Steve / Matt F) B) Print for an hour at speed to finisher with stapling (all) C) Copy for an hour **at speed** 35ppm (40ppm is at speed) **D) Enter/exit powersave (Steve / Mike)** Approved to push out to MM31 E) Mfg test suite exec on Coral **emulator with FIM support** (Terrese) **F) Automated FIM – no bash prompt (Steve)** Approved to push to MM31
4	Start porting CE code to ARM	A) Build single ARM system (Terrese) Feasibility proven. 2 DLL's to re-compile. B) High-level analysis of FW performance on ARM (Pat) Lowered priority.
5	Fleet Integr plan	Align on content for Coral/Garnet "slivers" of end-to-end agile test in ES. Overall plan in place. Need sliver details or will just deliver same as to PTO's.
BONUS		1st ARM/CE product - End-to-end boot, print, copy

■ Done ■ **Not done**

■ Close enough

FIGURE 6: ENTERPRISE-LEVEL CONTINUOUS IMPROVEMENT

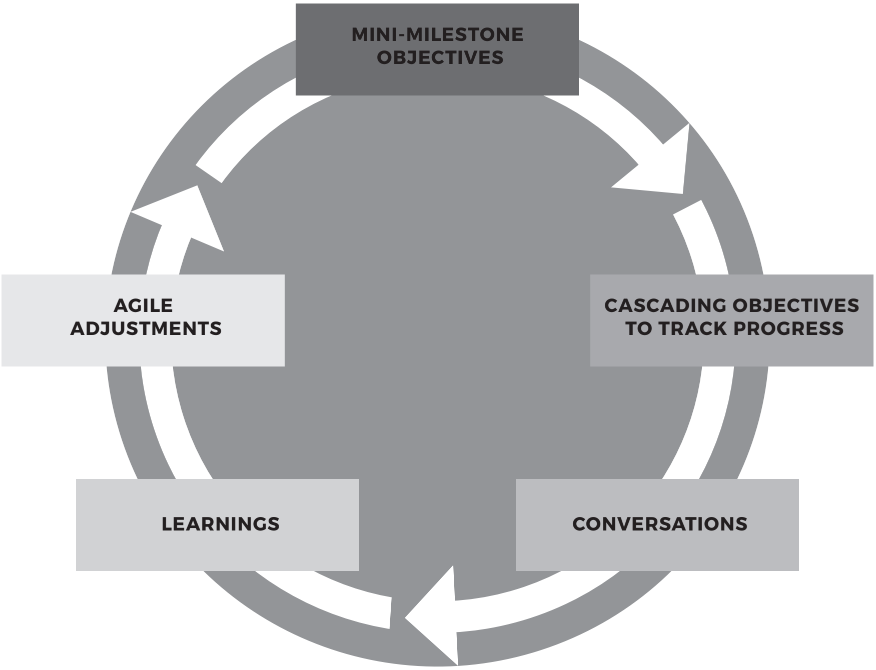

MINI-MILESTONE OBJECTIVES
CASCADING OBJECTIVES TO TRACK PROGRESS
CONVERSATIONS
LEARNINGS
AGILE ADJUSTMENTS

FIGURE 7: LONG-TERM PREDICTABILITY FOR SOFTWARE SCHEDULES

Do we really need the predictability of our current planning processes?
Are our current planning processes really that accurate?
50% of all software is never used or does not meet the business objectives!

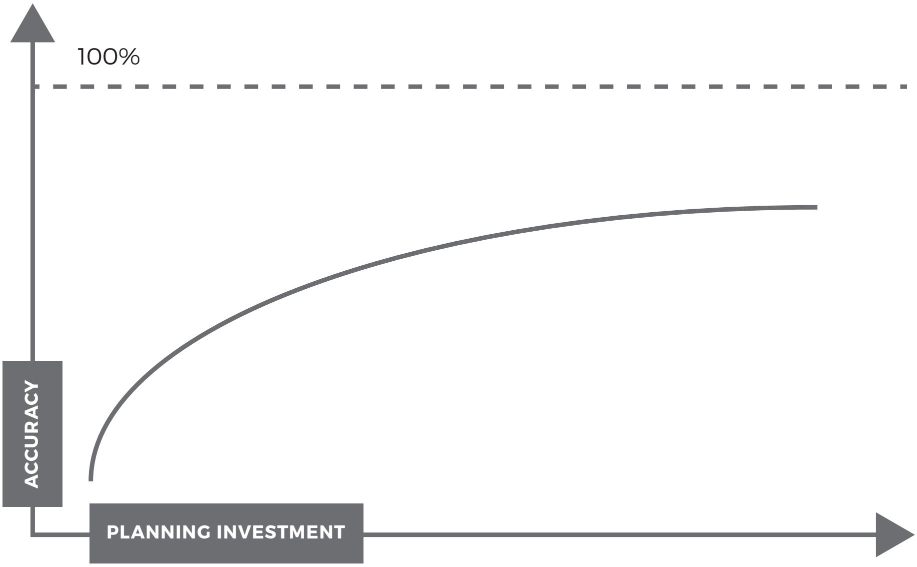


FIGURE 8: LEFTOVER CAPACITY

		HIGH-LEVEL ESTIMATE—FW ENGINEERING MONTHS												
RANK	INITIATIVE	Component 1 (25-30)	Component 2 (20-25)	Component 3 (30-40)	Component 4 (30-40)	Component 5 (20-30)	Component 6 (20-30)	Component 7 (20-30)	Component 8 (15-25)	Component 9 (20-30)	Component 10 (40-50)	Component 11 (20-30)	Other Items	TOTAL
1	Initiative A			21			5	3			1			30
2	Initiative B	3							4				17	24
3	Initiative C		5							1	2	1		9
4	Initiative D							10		2	2	2		16
5	Initiative E					20				3			5	28
6	Initiative F	23							5		6		2	36
7	Initiative G										2			2
8	Initiative H									5				5
9	Initiative I												3	3
10	Initiative J		20	**27**			17			**21**	**39**	17	9	150
11	Initiative K			**3**	30		3		3		**14**		12	65
12	Initiative L										**2**			2
13	Initiative M	**3**						10		**6**	**6**	6		31
		29	25	51	30	20	25	23	12	38	74	26	59	401

Time maxed-out

FIGURE 9: FUTURESMART FIRMWARE USER STORIES PER SPRINT

ESTIMATING SHORT-TERM FEATURES BASED ON DELIVERY

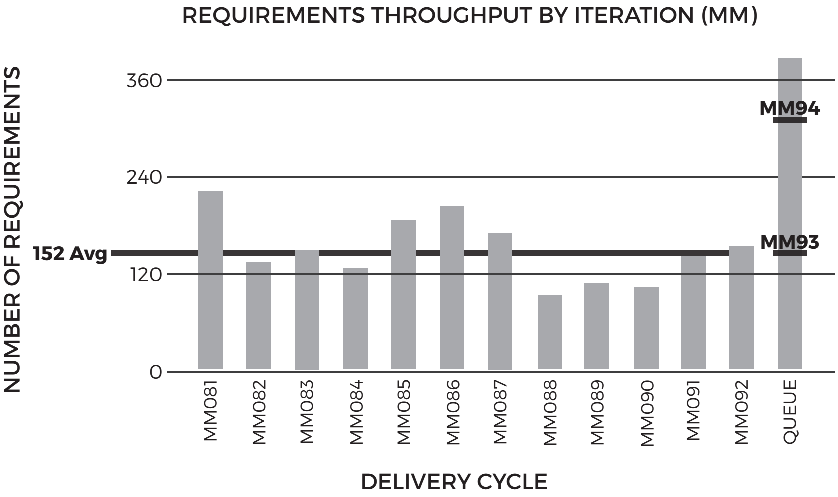


FIGURE 10: TRAIN WRECK BLOCKING CODE PROGRESSION

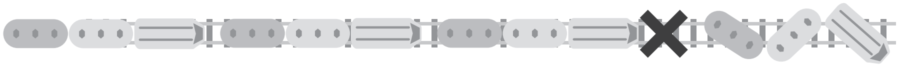


FIGURE 11: GATING COMMITS

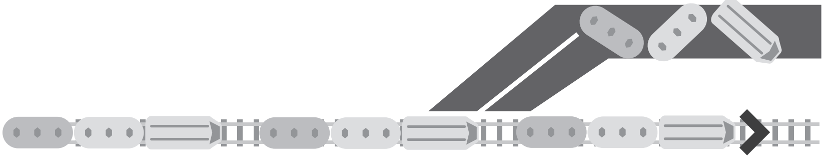


FIGURE 12: ENSURING THE ARCHITECTURE/BUILD SYSTEM IS READY

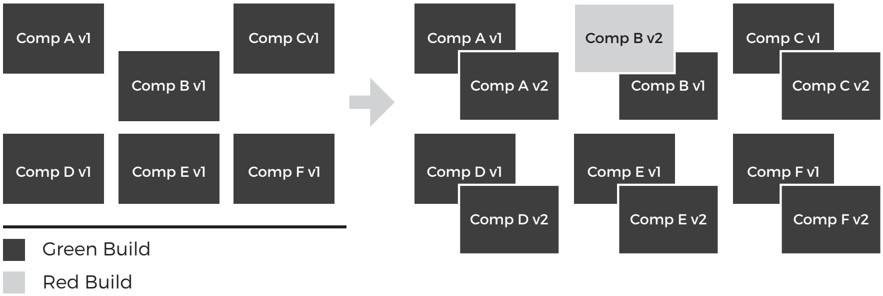


FIGURE 13: CONTINUOUS DELIVERY
PICKING THE RIGHT TOOL FOR THE JOB

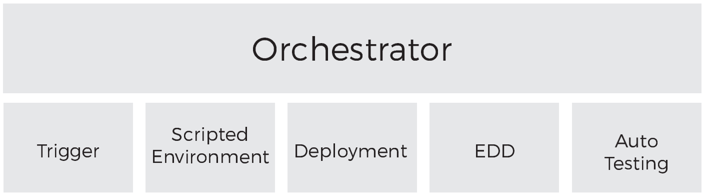


FIGURE 14: SCRIPTED ENVIRONMENT ARCHITECTURE

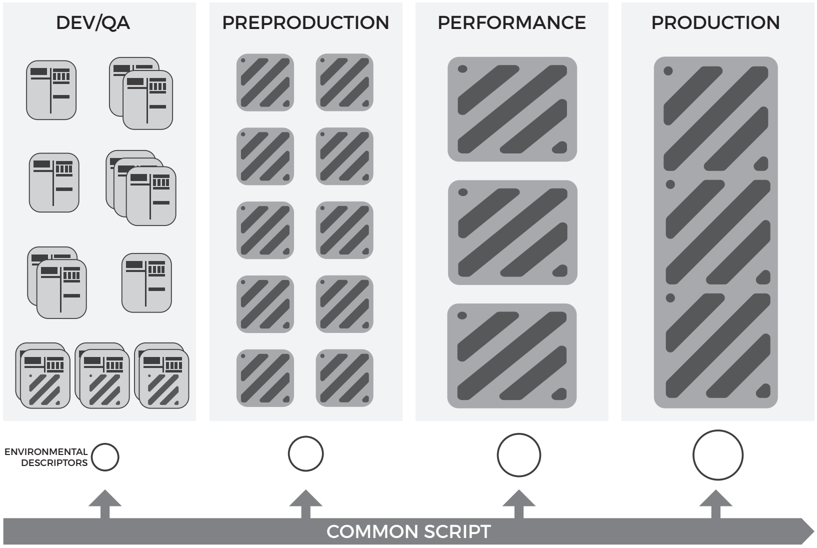
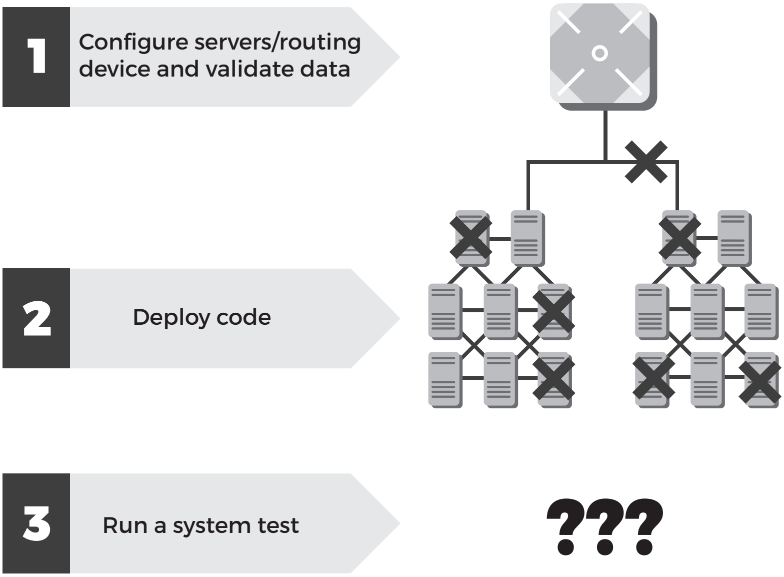


FIGURE 15A: FINDING THE OFFENDING COMPONENT QUICKLY IMPROVES PRODUCTIVITY

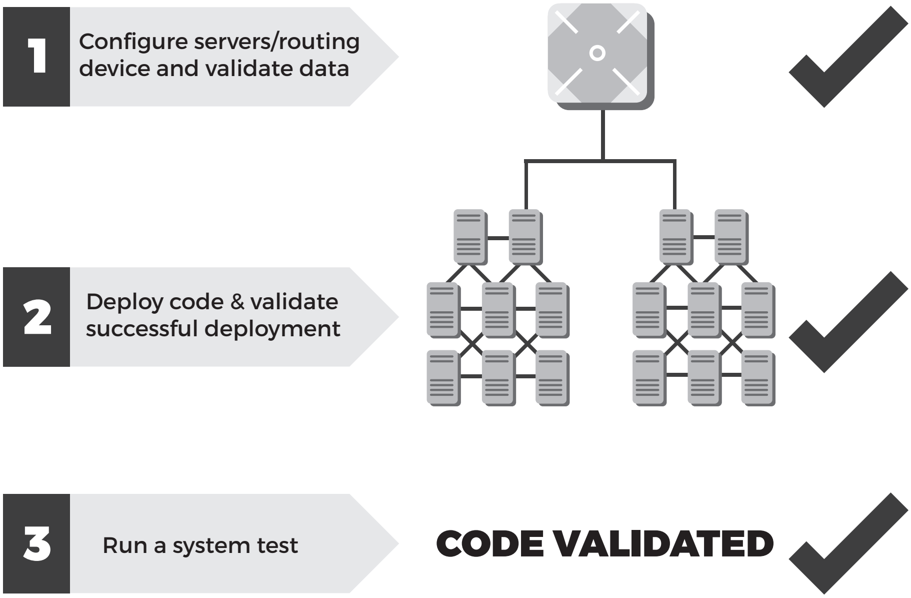


1 Configure servers/routing device and validate data

2 Deploy code

3 Run a system test

FIGURE 15B: FINDING THE OFFENDING COMPONENT QUICKLY IMPROVES PRODUCTIVITY



1 Configure servers/routing device and validate data ✓

2 Deploy code & validate successful deployment ✓

3 Run a system test

FIGURE 16: BUILDING UP THE ENTERPRISE SYSTEM

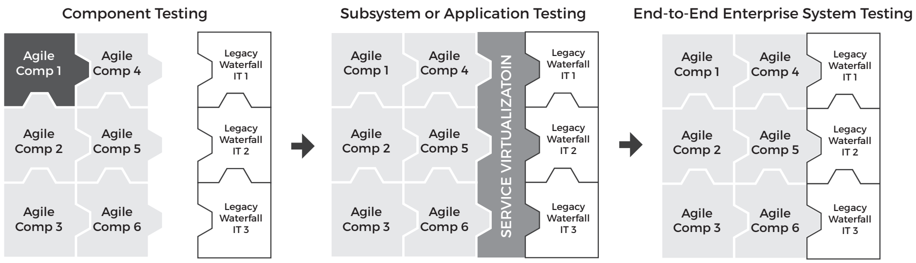


FIGURE 17: BUILDING UP THE ENTERPRISE SYSTEM

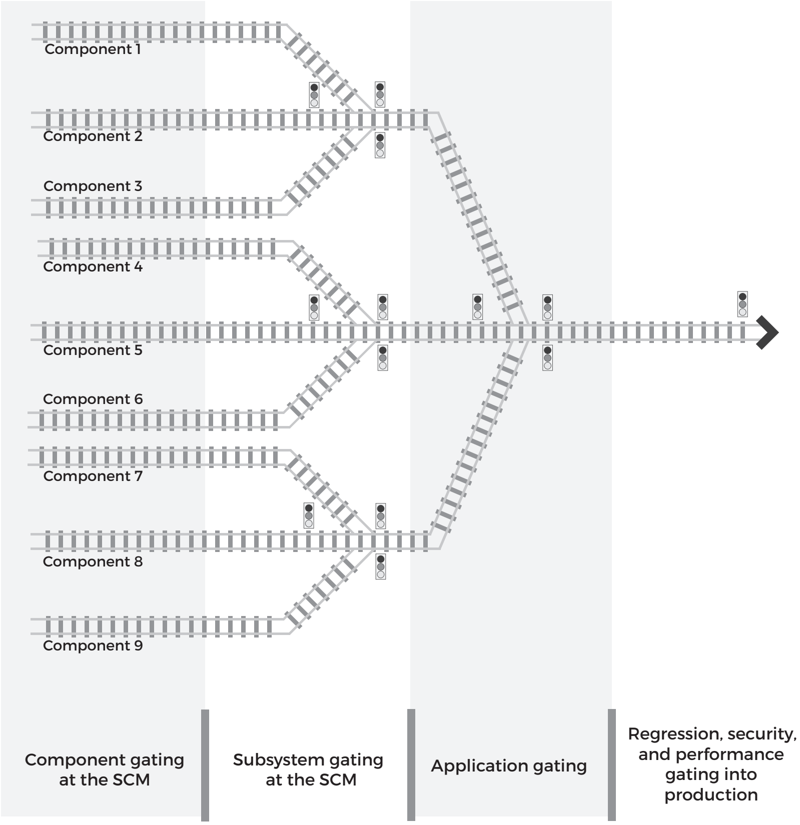


FIGURE 18: APPLICATION GATING

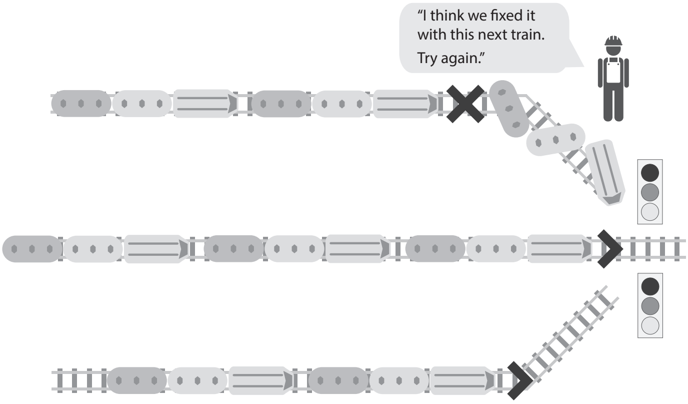


FIGURE 19: LOOSELY COUPLED ARCHITECTURE DEPLOYMENT PIPELINE

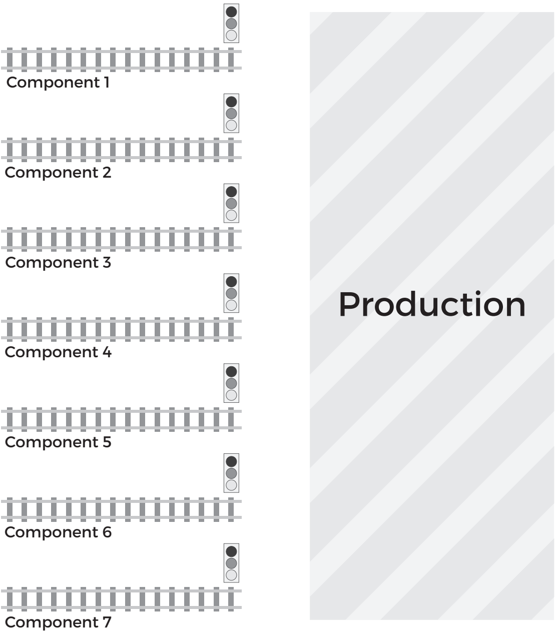


FIGURE 20: RELEASE READINESS

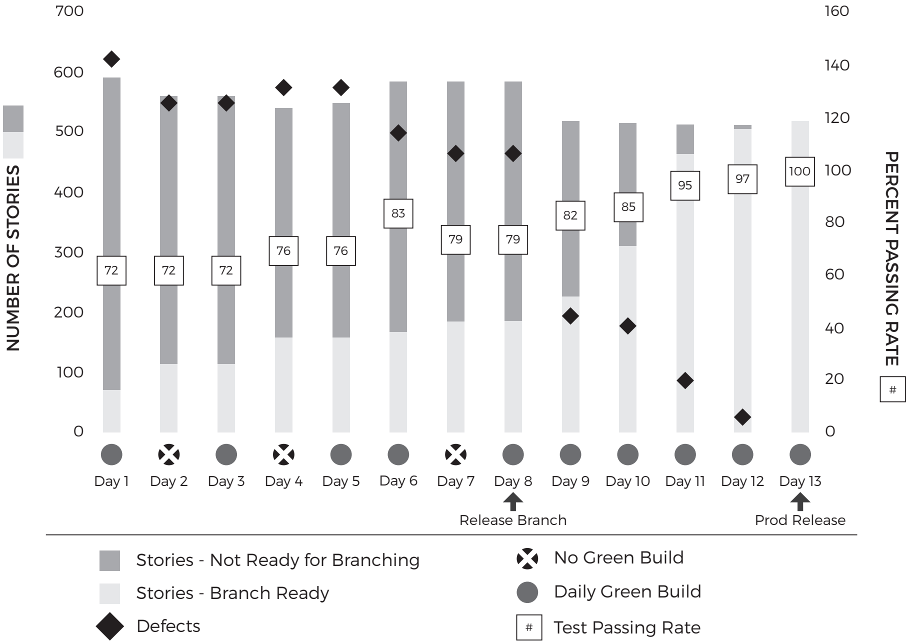


FIGURE 21: FARMING THE BUILD ACCEPTANCE

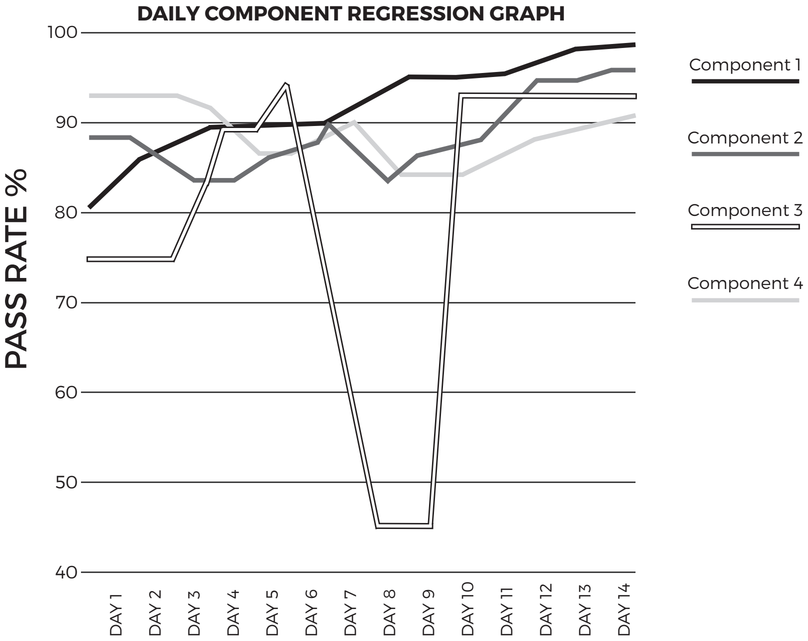


FIGURE 22: CONTINUOUS INTEGRATION AND TEST SYSTEM

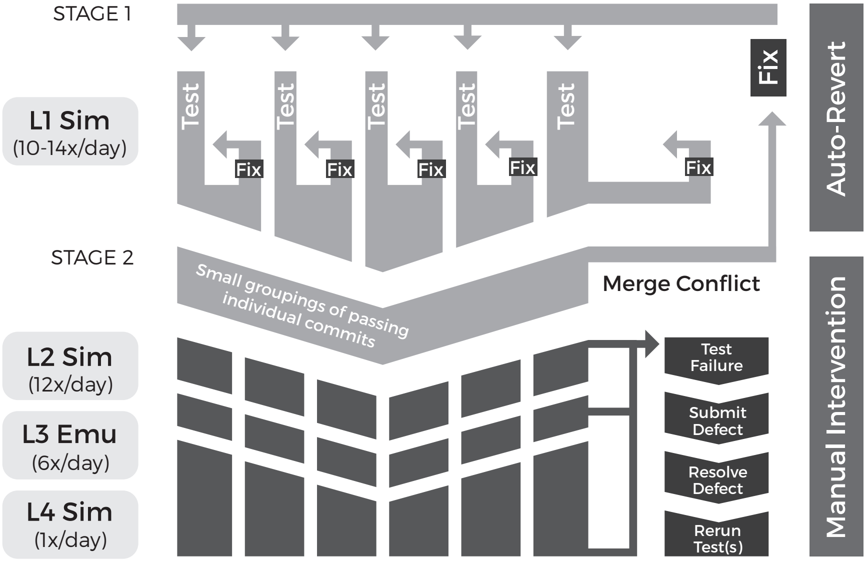


FIGURE 23: EMBEDDED SYSTEM PIPELINE

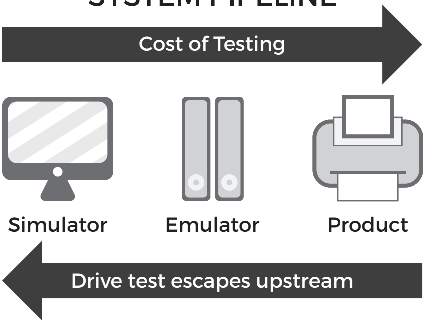
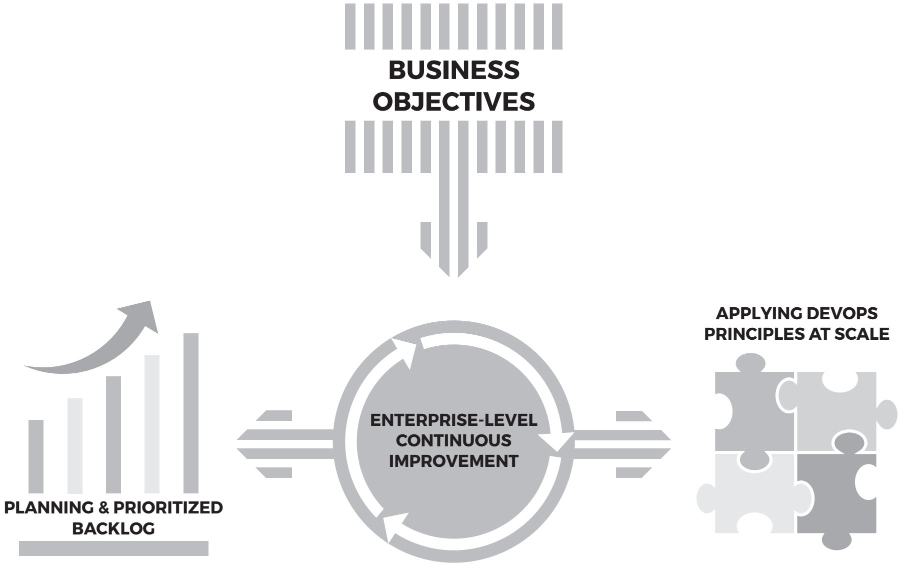


FIGURE 24: ENTERPRISE-LEVEL TRANSFORMATION


BUSINESS OBJECTIVES
PLANNING & PRIORITIZED BACKLOG
ENTERPRISE-LEVEL CONTINUOUS IMPROVEMENT
APPLYING DEVOPS PRINCIPLES AT SCALE

FURTHER READING

Lean Enterprise: How High Performance Organizations Innovate at Scale
Jez Humble, Joanne Molesky, and Barry O'Reilly

This is the most important book executives should be reading as soon as they have developed the ability to release on a more frequent basis. It shows how to take advantage of this new capability to address the 50% of features that are never used or do not meet their intended business objectives.

A Practical Approach to Large-Scale Agile Development
Gary Gruver, Mike Young, and Pat Fulghum

This is a good, easy-to-read case study that will give the reader a good feel for the transformation journey of one organization. It provides more details on the HP experience referenced heavily in this book.

Toyota Kata: Managing People for Improvement, Adaptiveness, and Superior Results
Mike Rother

This book provides a manufacturing example of how a continuous improvement culture can be a long-term competitive advantage. There are a lot of similarities to what is being done in software, but readers should be aware that software processes are also different from manufacturing. Work is unique each time instead of being repetitive, so it is going to more difficult to get a quantitative feel for each change. Software leaders are going to have to spend time in the organization getting a more qualitative feel for the effect of the change.

Cucumber & Cheese: A Tester's Workshop
Jeff Morgan

This book should be read by lead developers and lead testers to ensure you are creating an automated testing framework that is maintainable and that quickly localizes defects.

Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble and David Farley

This is a must-read for all your engineers working on Continuous Delivery.

Refactoring Databases: Evolutionary Database Design
Scott W. Ambler and Pramod J. Sadalage

This is must-read for all your database administrators and anyone telling you that trunk can't be always releasable due to database schema changes.

ABOUT THE AUTHORS

GARY GRUVER is an experienced executive with a proven track record of transforming software development processes and working with executives in large organizations. As coauthor of *A Practical Approach to Large-Scale Agile Development*, he documents how HP revolutionized software development while he was the director of the LaserJet Firmware development lab at HP. As VP of QE, Release, and Operations at Macys.com he led their transition to continuous delivery. Gary currently lives in Idaho with his wife and enjoys skiing, hiking, and mountain biking.

practicallargescaleagile.com
✉ gbgruver@gmail.com
🐦 @GRUVERGary

TOMMY MOUSER has been directly involved in the design, development, qualication, and delivery of software systems for over 30 years. For the past eight years, while working at HP and Macys.com, his primary focus has been on working with his teams and partners on a journey toward Agile methodologies. Tommy currently resides in Boise, Idaho, with his wife, Debbie. He enjoys a wide range of outdoor activities and has recently gotten into fly-fishing with some of his longtime friends and coworkers.

✉ tmouser@gmail.com

ACKNOWLEDGMENTS

Many people have contributed to this book. We would like to thank everyone we have worked with over the years that helped us better understand how to develop software. The ideas shared in this book are an accumulation of everything we have learned from working with each of you on a constant journey of improving software development processes. Without these discussions and debates, our understanding would not be as rich and the book would not be as complete.

We would like to thank everyone that has taken time to give us feedback on early versions of the book (in alphabetical order): Arun Dutta, Jim Highsmith, Jez Humble, Vinod Peris, Mary Poppendieck, Tom Poppendieck, and Lance Sleeper. Your input significantly improved the final product.

We would also like to thank the production staff at IT Revolution: Kate Sage, Robyn Crummer-Olson, Todd Sattersten, and Gene Kim. Your rigor and approach forced clarity of thought and crispness of message. We thank you, and our readers will benefit from a shorter and easier-to-read book.