

# SOFTWARE PROJECT SECRETS

WHY SOFTWARE PROJECTS FAIL

---

*George Stepanek*

Apress®

# *Software Project Secrets: Why Software Projects Fail*

Copyright © 2005 by George Stepanek

Lead Editor: Dominic Shakeshaft

Technical Reviewer: David Putnam

Editorial Board: Steve Anglin, Dan Appleman, Ewan Buckingham, Gary Cornell, Tony Davis,  
Jason Gilmore, Jonathan Hassell, Chris Mills, Dominic Shakeshaft, Jim Sumser

Associate Publisher: Grace Wong

Project Manager: Sofia Marchant

Copy Edit Manager: Nicole LeClerc

Copy Editor: Liz Welch

Assistant Production Director: Kari Brooks-Copony

Production Editor: Ellie Fountain

Compositor: Dina Quan

Proofreader: Patrick Vincent

Indexer: Carol Burbo

Artist: Kinetic Publishing Services, LLC

Interior and Cover Designer: Kurt Krames

Manufacturing Manager: Tom Deboliski

Library of Congress Cataloging-in-Publication Data

Stepanek, George.

Software project secrets : why software projects fail / George  
Stepanek.

p. cm.

ISBN 1-59059-550-5

1. Computer software--Development. 2. Project management. I. Title.

QA76.76.D47S733 2005

005.1--dc22

2005019810

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail [orders-ny@springer-sbm.com](mailto:orders-ny@springer-sbm.com), or visit <http://www.springeronline.com>.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone 510-549-5930, fax 510-549-5939, e-mail [info@apress.com](mailto:info@apress.com), or visit <http://www.apress.com>.

The information in this book is distributed on an "as is" basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

# Contents

---

ABOUT THE AUTHOR . . . . . xiii

ABOUT THE TECHNICAL REVIEWER . . . . . xv

ACKNOWLEDGMENTS . . . . . xvii

**PART I**  
**WHY SOFTWARE PROJECTS FAIL . . .**

**CHAPTER 1** INTRODUCTION . . . . . 3

**CHAPTER 2** WHY SOFTWARE IS DIFFERENT . . . . . 7

    1. Software Is Complex . . . . . 8

    2. Software Is Abstract . . . . . 10

    3. Requirements Are Incomplete . . . . . 11

    4. Technology Changes Rapidly . . . . . 12

    5. Best Practices Are Not Mature . . . . . 13

    6. Technology Is a Vast Domain . . . . . 15

    7. Technology Experience Is Incomplete . . . . . 16

    8. Software Development Is Research . . . . . 16

    9. Repetitive Work Is Automated . . . . . 18

    10. Construction Is Actually Design . . . . . 19

    11. Change Is Considered Easy . . . . . 20

    12. Change Is Inevitable . . . . . 21

    Summary . . . . . 22

**CHAPTER 3** PROJECT MANAGEMENT ASSUMPTIONS . . . . . 23

Hidden Assumptions . . . . . 24

    The PMBOK . . . . . 24

Scope Management . . . . . 25

    When Should Scope Definition Occur? . . . . . 27

Time Management . . . . . 28

    Activity Definition . . . . . 28

    Activity Sequencing . . . . . 30

    Activity Duration Estimating . . . . . 34

    Schedule Development . . . . . 36

Cost Management . . . . . 37

    Resource Planning . . . . . 38

    Software Documentation . . . . . 39

    Developer Productivity . . . . . 42

    Cost Estimating . . . . . 43

Quality Management . . . . . 44

    Metrics . . . . . 44

    Checklists . . . . . 45

Risk Management . . . . . 45

    Risk Acceptance . . . . . 46

    Risk Transference . . . . . 47

    Risk Avoidance . . . . . 48

    Risk Mitigation . . . . . 48

Summary . . . . . 49

<b>CHAPTER 4</b>	<b>CASE STUDY: THE BILLING SYSTEM PROJECT . . . . .</b>	<b>51</b>
	Requirements . . . . .	51
	Planning . . . . .	52
	Design . . . . .	54
	Construction . . . . .	54
	Coding . . . . .	55
	Integration . . . . .	55
	Testing . . . . .	57
	Death March . . . . .	58
	Aftermath . . . . .	59
	Summary . . . . .	60

## PART II

### ... AND HOW TO MAKE THEM SUCCEED

<b>CHAPTER 5</b>	<b>THE NEW AGILE METHODOLOGIES . . . . .</b>	<b>65</b>
	Selected Methodologies . . . . .	66
	Other Agile Methodologies . . . . .	66
	Crystal . . . . .	67
	1. Frequent Delivery . . . . .	68
	2. Reflective Improvement . . . . .	69
	3. Close or Osmotic Communication . . . . .	70
	4. Personal Safety . . . . .	71
	5. Focus . . . . .	72
	6. Easy Access to Expert Users . . . . .	72
	7. Technical Environment with Automated Tests, Configuration Management, and Frequent Integration . . . . .	73
	Using Crystal . . . . .	74
	Extreme Programming . . . . .	75
	1. The Planning Game . . . . .	76
	2. Testing . . . . .	77
	3. Pair Programming . . . . .	78
	4. Refactoring . . . . .	78
	5. Simple Design . . . . .	79
	6. Collective Code Ownership . . . . .	80

7. Continuous Integration . . . . .	80
8. On-Site Customer . . . . .	81
9. Small Releases . . . . .	81
10. 40-Hour Week . . . . .	81
11. Coding Standards . . . . .	82
12. System Metaphor . . . . .	82
Using XP . . . . .	83
The Rational Unified Process . . . . .	84
Phases . . . . .	86
Iterations . . . . .	87
Roles . . . . .	87
Artifacts . . . . .	87
Activities and Workflows . . . . .	88
Process Configuration . . . . .	88
Use Case–Driven Development . . . . .	89
Visual Modeling . . . . .	89
Using RUP . . . . .	90
Mitigating Risks with Agility . . . . .	91
1. Incomplete Requirements and Scope Changes . . . . .	91
2. Tools and Technologies Don't Work As Expected . . . . .	92
3. Developers Lack Skills and Expertise . . . . .	92
4. The New Software Has Defects and Requires Rework . . . . .	92
5. Project Staff Turnover . . . . .	93
Summary . . . . .	94
<b>CHAPTER 6 BUDGETING AGILE PROJECTS . . . . .</b>	<b>97</b>
Budgeting for Software Development . . . . .	98
1. Continuous Development . . . . .	100
2. On-Demand Programming . . . . .	101
3. SWAT Teams . . . . .	102
4. Subteam Encapsulation . . . . .	104
5. Feature Trade-off . . . . .	106
6. Triage . . . . .	106
7. Scoping Studies . . . . .	108

	Combining These Techniques . . . . .	109
	Major Legacy System . . . . .	110
	Minor Legacy Application . . . . .	110
	Major New System . . . . .	111
	Minor New Application . . . . .	112
	Agile Offshore Outsourcing . . . . .	112
	Summary . . . . .	114
<b>CHAPTER 7</b>	<b>CASE STUDY: THE BILLING SYSTEM REVISITED . . . . .</b>	<b>115</b>
	Methodology . . . . .	115
	Inception . . . . .	116
	Scoping Study . . . . .	117
	Project Planning Meeting . . . . .	118
	Elaboration . . . . .	121
	Review Meeting . . . . .	122
	Construction . . . . .	123
	Construction Iteration 5 . . . . .	124
	Transition . . . . .	126
	Deployment . . . . .	126
	Aftermath . . . . .	127
	Summary . . . . .	128
<b>CHAPTER 8</b>	<b>AFTERWORD . . . . .</b>	<b>131</b>
	APPENDIX: THE AGILE MANIFESTO . . . . .	133
	Manifesto for Agile Software Development . . . . .	133
	GLOSSARY . . . . .	135
	BIBLIOGRAPHY . . . . .	143
	INDEX . . . . .	151