

# **SOFTWARE ENGINEERING**

*Planning for  
Change*

**David Alex Lamb**

TP31  
L218

8862699



E8862699

# Software Engineering: — Planning for Change —

**David Alex Lamb**

*Department of Computing and Information Science  
Queen's University  
Kingston, Ontario, Canada*



PRENTICE HALL, Englewood Cliffs, New Jersey 07632

LAMB, DAVID ALEX, 1954-  
Software engineering.

QA76.76.D47L36

Bibliography: p.  
Includes index.

1. Computer software—Development. I. Title.  
QA76.76.D47L36 1988 005.1 87-14321  
ISBN 0-13-822982-1

Editorial/production supervision  
and interior design: *Mary Jo Stanley*  
Cover design: *Lundgren Graphics, Ltd.*  
Manufacturing buyer: *Gordon Osbourne*



© 1988 by Prentice Hall  
A Division of Simon & Schuster  
Englewood Cliffs, New Jersey 07632

All rights reserved. No part of this book may be  
reproduced, in any form or by any means,  
without permission in writing from the publisher.

Printed in the United States of America  
10 9 8 7 6 5 4 3 2 1

ISBN 0-13-822982-1 025

Prentice-Hall International (UK) Limited, *London*  
Prentice-Hall of Australia Pty. Limited, *Sydney*  
Prentice-Hall Canada Inc., *Toronto*  
Prentice-Hall Hispanoamericana, S.A., *Mexico*  
Prentice-Hall of India Private Limited, *New Delhi*  
Prentice-Hall of Japan, Inc., *Tokyo*  
Simon & Schuster Asia Pte. Ltd., *Singapore*  
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

# Software Engineering: Planning for Change



8862699

*In loving memory*

Roy William Lamb  
1905-1986

Richard M. Beard  
1924-1986



# Preface

---

This book introduces the fundamental ideas of software engineering. As with most terms naming broad categories, “software engineering” describes a field with a solid core of key ideas, along with fuzzy boundaries separating it from other fields. The Introduction (Chapter 1) outlines what I mean by the term; the Retrospective (Chapter 20) reflects further on the key ideas and mentions other topics I have left out.

Most of the book requires only general familiarity with programming. You can understand some of the motivations better if you have written moderate-sized (1,000 to 3,000 line) programs in a higher-level language. Many programming examples presume knowledge of Pascal; some use Ada or C. Chapter 6 requires some familiarity with first-order logic. Part III requires some mathematical sophistication; you should be familiar with discrete mathematics, particularly set theory, functions, and mathematical logic.

Although this textbook should be valuable to anyone who wishes to understand software engineering, I have aimed it primarily at senior-level undergraduates or graduate students who have had little or no experience working with others on a large program. Thus I include some material, such as that of Chapter 15, that should be familiar to those who have already had some form of programming job. Furthermore, I believe the only way to teach software engineering is to have students carry out a moderate-sized group project; this text is geared to a one-semester or one-year group project course.

I could not have written this book without the guidance of David Parnas, who has taught software engineering project courses for many years. I based much of the material of Part II on a collection of papers he put together for his course at the University of Victoria. The bibliography lists other sources I consulted. Other material comes from my experience as a member of the technical staff at Bell-Northern Research, and a staff scientist at Tartan Laboratories. While I was building and maintaining systems as a Research Assistant at Carnegie-Mellon University, I learned much from Ivor Durham, Craig Everhart, Joe Newcomer, Brian Reid, Tom Rodeheffer, and Steve Shafer. Some of the material for Chapter 17 came from discussions with Ed Satterthwaite about system modeling, and with Ellen Borison about her Ph.D. dissertation.

Thanks to (in alphabetical order) Richard Beard, John Nestor, Joe Newcomer, and Sid Penstone for enlightening discussions of what it means to be an engineer. Margaret Lamb proofread two earlier drafts of the book, wrote the program that produced the index, suggested some of the projects in Chapter 4, discovered several embarrassing mistakes in examples, and helped me to improve the trace specifications in Appendix E. Thanks also to the students in CISC422 and CISC838 at Queen's University during 1985 and 1986, who endured my efforts to write this book; Phil Beaudet, Susan Lee, Karen Lefave, and Jim Roche helped by commenting on earlier drafts of the manuscript.

---

# Contents

---

**Preface   xxi**

**Part I:   Overview**

**1   Introduction   1**

- 1.1   Relation to Other Fields   2
- 1.2   Why the Difficulty?   4
- 1.3   Using This Book   4
- Further Reading   6

**2   The Lifetime of a Software System   7**

- 2.1   Typical Activities   7
  - 2.1.1   Opportunity Study   8
  - 2.1.2   Problem Formulation   9
  - 2.1.3   Product Realization   10
  - 2.1.4   Delivery and Beyond   10
- 2.2   Project Documents   10
- 2.3   Overlap and Cycles   11
- 2.4   Personnel   12
- Further Reading   12



### 3 Technical Writing 13

- 3.1 Planning a Document 13
- 3.2 Organizing a Document 15
- 3.3 Proofreading 16
- Further Reading 17
- Exercises 17

## Part II: Software Lifetime

### 4 Requirements Analysis and Specification 19

- 4.1 What Is a Requirement? 19
- 4.2 Problem Analysis 21
  - 4.2.1 Dealing with Customers 21
  - 4.2.2 Why Build This System? 22
  - 4.2.3 Requirements Considerations 23
  - 4.2.4 Sketching Initial Requirements 24
- 4.3 The Requirements Specification Document 26
  - 4.3.1 Document Organization 28
  - 4.3.2 Input and Output 30
  - 4.3.3 Describing Software Functions 31
    - 4.3.3.1 Modes 32
    - 4.3.3.2 Concurrent Systems 32
- 4.4 User Documentation 33
- 4.5 Designing Subsets 35
- 4.6 Validating Requirements 37
- Further Reading 37
- Exercises 37
- Project Exercises 37

### 5 Preliminary Design 41

- 5.1 System Structure 41
  - 5.1.1 The *Is Composed Of* Relation 42
  - 5.1.2 The *Uses* Relation 43
- 5.2 Desiderata for Modular Systems 44
  - 5.2.1 Program Families 45
  - 5.2.2 Stepwise Refinement 46
- 5.3 Information Hiding 47
  - 5.3.1 Hiding Representations 48
  - 5.3.2 Hiding Policies 50
  - 5.3.3 Hiding Time of Operations 51
- 5.4 The Module Decomposition Document 51
  - 5.4.1 System Modules 52

- 5.4.2 Behavior Modules 53
- 5.4.3 Software Decision Modules 53
- 5.4.4 How to Modularize 54
- 5.5 Module Dependencies 56
- 5.6 Relation to Other Methods 56
- Further Reading 57
- Exercises 58
- Project Exercises 58

## **6 Module Interfaces 59**

- 6.1 Deciding on Functionality 59
  - 6.1.1 Abstract Interfaces 60
  - 6.1.2 Imports and Exports 60
- 6.2 Information-Hiding Complexities 62
  - 6.2.1 Iterators 63
  - 6.2.2 Accumulators 67
  - 6.2.3 Path Expressions 70
  - 6.2.4 Editors 71
- 6.3 What Is a Specification? 72
  - 6.3.1 Using Specifications 72
  - 6.3.2 Abstract Programs 73
- 6.4 Defining Interface Procedures 75
  - 6.4.1 Specification Content 76
  - 6.4.2 Undefined Values 78
  - 6.4.3 Implementation Languages 79
  - 6.4.4 Specifying Input/Output Operations 80
  - 6.4.5 Summary Information 82
- 6.5 Restricted Interfaces 83
- Further Reading 84
- Exercises 84
- Project Exercises 85

## **7 Module Implementation 87**

- 7.1 Representing Modules 87
  - 7.1.1 No Module Facilities 88
  - 7.1.2 Separate Compilation 89
  - 7.1.3 Module Constructs 91
- 7.2 Casting Specifications into Implementations 91
  - 7.2.1 Interface Procedures 92
  - 7.2.2 Namespace Control 94
  - 7.2.3 Undesired Event Handling 94
  - 7.2.4 Undefined Values 100
  - 7.2.5 Initialization 101

7.3	Module Design	101
7.4	Coding	102
7.4.1	Style Conventions	103
7.4.2	Commenting Conventions	104
7.4.3	Usage Conventions	106
	Exercises	107
	Project Exercises	108
<b>8</b>	<b>Testing</b>	<b>109</b>
8.1	Scaffolding	111
8.1.1	Test Drivers	111
8.1.2	Stubs	111
8.2	Unit Testing	113
8.3	Integration Testing	114
8.3.1	Styles of Integration	114
8.3.2	Unit Testing versus Integration Testing	115
8.4	System Testing	115
8.5	Regression Testing	116
8.6	Personnel	117
	Exercises	117
	Project Exercises	118
<b>9</b>	<b>System Delivery</b>	<b>119</b>
9.1	Site Preparation	119
9.2	User Training	120
9.3	System Introduction	121
9.4	The Effect of Market Size	122
<b>10</b>	<b>Evolution</b>	<b>125</b>
10.1	Categories of Maintenance	125
10.2	Levels of Support	126
10.2.1	Full Support	126
10.2.2	The Worst Case	127
10.2.3	Restructuring	127
10.3	Product Support	129
10.3.1	Problem Reports	129
10.3.2	Multiple Releases	130
10.3.3	Distributing Information	131
10.3.4	Conversion Support	131
	Further Reading	132
	Project Exercises	132

## **Part III: Specifications and Verification**

### **11 Introduction to Specifications 133**

- 11.1 Common Issues 133
- 11.2 Verification 134
- 11.3 Limitations of Specifications 134
- Further Reading 136

### **12 Algebraic Specifications 137**

- 12.1 Outline of a Specification 137
  - 12.1.1 Exceptions 139
  - 12.1.2 Constructor Functions 139
  - 12.1.3 Auxiliary Functions 140
- 12.2 Dealing with Side Effects 140
- 12.3 Reasoning About Algebraic Specifications 143
- 12.4 Advantages and Disadvantages 144
- Exercises 145

### **13 Trace Specifications 147**

- 13.1 State Machine Specifications 147
- 13.2 Trace Specifications 148
- 13.3 More Complex Traces 151
- 13.4 Normal Forms for Traces 153
- 13.5 Advantages and Disadvantages 155
- Further Reading 157
- Exercises 157

### **14 Abstract Modeling 159**

- 14.1 Overview of Verification 159
- 14.2 Overview of Abstract Modeling 160
- 14.3 Relation to Algebraic Specifications 162
- Further Reading 166
- Exercises 166

## **Part IV: Other Topics**

### **15 The Workplace 169**

- 15.1 Team Structure 169
  - 15.1.1 Small Teams 170
  - 15.1.2 Chief Programmer Teams 171
  - 15.1.3 Chief Programmer/Chief Engineer 172
  - 15.1.4 Designer/Team Leader 172

15.2	Meetings	172
15.2.1	Status Meetings	173
15.2.2	Brainstorming Sessions	173
15.2.3	Decision-Making Meetings	174
15.3	Programmer Psychology	174
15.3.1	Ego	175
15.3.2	Programmer Variability	175
15.3.3	Group Psychology	176
15.4	Personal Logs	176
	Further Reading	177
<b>16</b>	<b>Scheduling and Budgeting</b>	<b>179</b>
16.1	Schedules	179
16.2	Budgets	181
16.3	Estimating Time	182
	Further Reading	183
	Exercises	184
<b>17</b>	<b>Configuration Management</b>	<b>185</b>
17.1	The Need for Configuration Management	185
17.2	Configuration Identification	186
17.2.1	Objects	186
17.2.2	Relationships	187
17.3	Configuration Control	188
17.3.1	Baselines and Updates	189
17.3.2	Change Control	190
17.3.3	System Rebuilding	190
17.4	Configuration Status and Accounting	191
17.5	Configuration Hierarchy	192
	Further Reading	192
<b>18</b>	<b>Quality Assurance</b>	<b>193</b>
18.1	Measures of Quality	193
18.2	Validation	194
18.3	Reviews	195
18.3.1	Requirements Review	196
18.3.2	Design Review	197
18.3.3	Code Walkthroughs	198
	Exercises	199
	Project Exercises	199

**19 Tools 201**

- 19.1 Evaluating a Tool 201
- 19.2 Software Engineering Environments 203
- 19.3 Reusable Components 203
  - 19.3.1 Module Libraries 204
  - 19.3.2 Program Generators 204
- 19.4 Prototyping 205
- 19.5 Toolsmith and Tools Group 206
- Further Reading 207

**20 Retrospective 209**

- 20.1 Fundamental Themes 209
- 20.2 What Else Is There? 210
- 20.3 Do People Really Do This? 210
- 20.4 A Personal View 212

**Appendices****A Sample User's Guide 215**

- A.1 Introduction 215
- A.2 Commands 216
  - A.2.1 Add Command 216
  - A.2.2 Change Command 217
  - A.2.3 Delete Command 217
  - A.2.4 Exit and Quit Commands 218
  - A.2.5 Help Command 218
  - A.2.6 What Command 218
  - A.2.7 Who Command 219
- A.3 Exception Summary 219
  - A.3.1 Add Command Exceptions 219
  - A.3.2 Input File Exceptions 220
  - A.3.3 Other Command Exceptions 221
  - A.3.4 Unchecked Problems 221
- A.4 Glossary 221

**B Sample Life-Cycle Considerations 223**

- B.1 Introduction 223
- B.2 Fundamental Assumptions 223
- B.3 Potential Changes 224
- B.4 Subsets 225

**C Sample System Test Plan 227**

- C.1 Introduction 227
- C.2 Functionality Tests 229
  - C.2.1 Basic Functionality 229
  - C.2.2 Advanced Functionality 230
- C.3 Exception Tests 230
  - C.3.1 Input File Tests 230
  - C.3.2 Capacity Exceptions 233

**D Sample Module Decomposition and Dependencies 235**

- D.1 Introduction 235
- D.2 System Module 236
  - D.2.1 Char Module 236
- D.3 Requirements Module 236
  - D.3.1 Add Module 236
  - D.3.2 Change Module 236
  - D.3.3 Delete Module 236
  - D.3.4 Command Module 236
  - D.3.5 GenFile Module 236
  - D.3.6 What Module 236
  - D.3.7 Who Module 236
- D.4 Software Decision Module 237
  - D.4.1 Find Module 237
  - D.4.2 Name Module 237
  - D.4.3 Person Module 237
  - D.4.4 RelDef Module 237
  - D.4.5 Set Module 237
  - D.4.6 Table Module 237
  - D.4.7 Word Module 237
- D.5 Module Dependencies 238
  - D.5.1 Add Module 239
  - D.5.2 Change Module 239
  - D.5.3 Char Module 239
  - D.5.4 Delete Module 239
  - D.5.5 Command Module 239
  - D.5.6 Find Module 239
  - D.5.7 GenFile Module 239
  - D.5.8 Main Module 239
  - D.5.9 Name Module 240
  - D.5.10 Person Module 240
  - D.5.11 RelDef Module 240
  - D.5.12 Set Module 240
  - D.5.13 Table Module 240

- D.5.14 What Module 240
- D.5.15 Who Module 240
- D.5.16 Word Module 240

## **E Sample Module Specifications 241**

- E.1 Introduction 241
- E.2 Command Module 242
- E.3 Find Module 242
- E.4 GenFile Module 242
- E.5 Name Module 243
- E.6 Person Module 245
- E.7 Person Module, Find Interface 248
- E.8 RelDef Module 249
- E.9 Set Module 251
  - E.9.1 Syntax 251
  - E.9.2 Auxiliary Functions 251
  - E.9.3 Equations 251
- E.10 What Module 252
- E.11 Who Module 252

## **F Sample Integration Test Plan 253**

- F.1 Introduction 253
- F.2 Unit Testing 254
- F.3 Integration Testing 254
- F.4 Test Drivers 255
  - F.4.1 Char Module 255
  - F.4.2 Command Module 255
  - F.4.3 Find Module 255
  - F.4.4 GenFile Module 255
  - F.4.5 Name Module 255
  - F.4.6 Person Module 256
  - F.4.7 RelDef Module 256
  - F.4.8 Set Module 256
  - F.4.9 Table Module 256
  - F.4.10 What Module 256
  - F.4.11 Who Module 257
  - F.4.12 Word Module 257
- F.5 Stubs 257
  - F.5.1 Person Module 257
  - F.5.2 What Module 257
  - F.5.3 Who Command 257



**G Sample Module Implementation Summary 259**

- G.1 Introduction 259
- G.2 Find Module 259
- G.3 GenFile Module 260
- G.4 Name Module 260
- G.5 RelDef Module 262
- G.6 Set Module 263
- G.7 Who Module 264
- G.8 Word Module 264

**H Sample Listing 265**

- H.1 Explanation of the Listing 265
- H.2 The Listing 267
- H.3 Code Walkthrough Report 273
  - H.3.1 Observations 273
  - H.3.2 Action Items 273

**I Sample Release Notice 275**

- I.1 Introduction 275
- I.2 Restrictions 275
- I.3 Input Files 275
- I.4 Exception Messages 276

**Glossary 277****Bibliography 279****Index 283**