

DEVELOPING SAFETY-CRITICAL SOFTWARE

A Practical Guide for Aviation Software and DO-178C Compliance

LEANNA RIERSON



CRC Press

Taylor & Francis Group
Boca Raton London New York

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

Contents

Preface.....	xxiii
Acknowledgments	xxv
Author	xxvii

Part I Introduction

1. Introduction and Overview	3
Acronyms	3
1.1 Defining Safety-Critical Software	3
1.2 Importance of Safety Focus	4
1.3 Book Purpose and Important Caveats	6
1.4 Book Overview	8
References	9

Part II Context of Safety-Critical Software Development

2. Software in the Context of the System.....	13
Acronyms	13
2.1 Overview of System Development	13
2.2 System Requirements	16
2.2.1 Importance of System Requirements	16
2.2.2 Types of System Requirements	16
2.2.3 Characteristics of Good Requirements	17
2.2.4 System Requirements Considerations	19
2.2.4.1 Integrity and Availability Considerations	19
2.2.4.2 Other System Requirements Considerations	20
2.2.5 Requirements Assumptions	23
2.2.6 Allocation to Items	23
2.3 System Requirements Validation and Verification	23
2.3.1 Requirements Validation	23
2.3.2 Implementation Verification	24
2.3.3 Validation and Verification Recommendations	24
2.4 Best Practices for Systems Engineers	27
2.5 Software’s Relationship to the System	30
References	31

- 3. Software in the Context of the System Safety Assessment 33
 - Acronyms 33
 - 3.1 Overview of the Aircraft and System Safety Assessment Process 33
 - 3.1.1 Safety Program Plan..... 34
 - 3.1.2 Functional Hazard Assessment..... 35
 - 3.1.3 System Functional Hazard Assessment 37
 - 3.1.4 Preliminary Aircraft Safety Assessment..... 37
 - 3.1.5 Preliminary System Safety Assessment 38
 - 3.1.6 Common Cause Analysis 38
 - 3.1.7 Aircraft and System Safety Assessments 40
 - 3.2 Development Assurance 40
 - 3.2.1 Development Assurance Levels..... 41
 - 3.3 How Does Software Fit into the Safety Process? 43
 - 3.3.1 Software’s Uniqueness 43
 - 3.3.2 Software Development Assurance 43
 - 3.3.3 Other Views 44
 - 3.3.4 Some Suggestions for Addressing Software in the System Safety Process 46
 - References 47

**Part III Developing Safety-Critical Software
 Using DO-178C**

- 4. Overview of DO-178C and Supporting Documents 51
 - Acronyms 51
 - 4.1 History of DO-178 51
 - 4.2 DO-178C and DO-278A Core Documents..... 55
 - 4.2.1 DO-278A and DO-178C Differences 57
 - 4.2.2 Overview of the DO-178C Annex A Objectives Tables 62
 - 4.3 DO-330: Software Tool Qualification Considerations 67
 - 4.4 DO-178C Technology Supplements 68
 - 4.4.1 DO-331: Model-Based Development Supplement 68
 - 4.4.2 DO-332: Object-Oriented Technology Supplement 69
 - 4.4.3 DO-333: Formal Methods Supplement 70
 - 4.5 DO-248C: Supporting Material 70
 - References 71
- 5. Software Planning 73
 - Acronyms 73
 - 5.1 Introduction 73
 - 5.2 General Planning Recommendations 74

- 5.3 Five Software Plans..... 78
 - 5.3.1 Plan for Software Aspects of Certification..... 78
 - 5.3.2 Software Development Plan..... 81
 - 5.3.3 Software Verification Plan..... 83
 - 5.3.4 Software Configuration Management Plan 86
 - 5.3.5 Software Quality Assurance Plan 89
- 5.4 Three Development Standards 90
 - 5.4.1 Software Requirements Standards..... 91
 - 5.4.2 Software Design Standards..... 92
 - 5.4.3 Software Coding Standards..... 94
- 5.5 Tool Qualification Planning 95
- 5.6 Other Plans 95
 - 5.6.1 Project Management Plan..... 95
 - 5.6.2 Requirements Management Plan 95
 - 5.6.3 Test Plan 95
- References 96

- 6. Software Requirements 97**
 - Acronyms..... 97
 - 6.1 Introduction 97
 - 6.2 Defining Requirement..... 98
 - 6.3 Importance of Good Software Requirements..... 99
 - 6.3.1 Reason 1: Requirements Are the Foundation
for the Software Development.....99
 - 6.3.2 Reason 2: Good Requirements Save Time
and Money 101
 - 6.3.3 Reason 3: Good Requirements Are Essential
to Safety 102
 - 6.3.4 Reason 4: Good Requirements Are Necessary
to Meet the Customer Needs..... 102
 - 6.3.5 Reason 5: Good Requirements Are Important
for Testing 102
 - 6.4 The Software Requirements Engineer..... 103
 - 6.5 Overview of Software Requirements Development..... 104
 - 6.6 Gathering and Analyzing Input to the Software Requirements ... 107
 - 6.6.1 Requirements Gathering Activities..... 107
 - 6.6.2 Requirements Analyzing Activities..... 108
 - 6.7 Writing the Software Requirements 109
 - 6.7.1 Task 1: Determine the Methodology..... 109
 - 6.7.2 Task 2: Determine the Software Requirements
Document Layout 111
 - 6.7.3 Task 3: Divide Software Functionality
into Subsystems and/or Features..... 112
 - 6.7.4 Task 4: Determine Requirements Priorities 112

6.7.5	A Brief Detour (Not a Task): Slippery Slopes to Avoid	113
6.7.5.1	Slippery Slope #1: Going to Design Too Quickly	113
6.7.5.2	Slippery Slope #2: One Level of Requirements	114
6.7.5.3	Slippery Slope #3: Going Straight to Code	114
6.7.6	Task 5: Document the Requirements	115
6.7.6.1	Document Functional Requirements	115
6.7.6.2	Document Nonfunctional Requirements	117
6.7.6.3	Document Interfaces	118
6.7.6.4	Uniquely Identify Each Requirement.....	119
6.7.6.5	Document Rationale	119
6.7.6.6	Trace Requirements to Their Source	120
6.7.6.7	Identify Uncertainties and Assumptions	120
6.7.6.8	Start a Data Dictionary	121
6.7.6.9	Implement Characteristics of Good Requirements	121
6.7.7	Task 6: Provide Feedback on the System Requirements	122
6.8	Verifying (Reviewing) Requirements	123
6.8.1	Peer Review Recommended Practices	125
6.9	Managing Requirements	128
6.9.1	Basics of Requirements Management.....	128
6.9.2	Requirements Management Tools	129
6.10	Requirements Prototyping	131
6.11	Traceability.....	132
6.11.1	Importance and Benefits of Traceability.....	132
6.11.2	Bidirectional Traceability.....	133
6.11.3	DO-178C and Traceability.....	135
6.11.4	Traceability Challenges.....	136
	References	138
	Recommended Readings	139
7.	Software Design	141
	Acronyms.....	141
7.1	Overview of Software Design.....	141
7.1.1	Software Architecture.....	142
7.1.2	Software Low-Level Requirements.....	142
7.1.3	Design Packaging	145
7.2	Approaches to Design	145
7.2.1	Structure-Based Design (Traditional).....	145
7.2.2	Object-Oriented Design	147
7.3	Characteristics of Good Design	148
7.4	Design Verification.....	153
	References	154

8. Software Implementation: Coding and Integration.....	157
Acronyms.....	157
8.1 Introduction.....	157
8.2 Coding.....	158
8.2.1 Overview of DO-178C Coding Guidance.....	158
8.2.2 Languages Used in Safety-Critical Software.....	159
8.2.2.1 Assembly Language.....	159
8.2.2.2 Ada.....	161
8.2.2.3 C.....	161
8.2.3 Choosing a Language and Compiler.....	162
8.2.4 General Recommendations for Programming.....	164
8.2.5 Special Code-Related Topics.....	176
8.2.5.1 Coding Standards.....	176
8.2.5.2 Compiler-Supplied Libraries.....	177
8.2.5.3 Autocode Generators.....	177
8.3 Verifying the Source Code.....	178
8.4 Development Integration.....	179
8.4.1 Build Process.....	179
8.4.2 Load Process.....	181
8.5 Verifying the Development Integration.....	181
References.....	182
Recommended Reading.....	182
 9. Software Verification.....	 185
Acronyms.....	185
9.1 Introduction.....	186
9.2 Importance of Verification.....	186
9.3 Independence and Verification.....	187
9.4 Reviews.....	189
9.4.1 Software Planning Review.....	189
9.4.2 Software Requirements, Design, and Code Reviews.....	190
9.4.3 Test Data Reviews.....	190
9.4.4 Review of Other Data Items.....	190
9.5 Analyses.....	190
9.5.1 Worst-Case Execution Time Analysis.....	192
9.5.2 Memory Margin Analysis.....	192
9.5.3 Link and Memory Map Analysis.....	193
9.5.4 Load Analysis.....	194
9.5.5 Interrupt Analysis.....	194
9.5.6 Math Analysis.....	194
9.5.7 Errors and Warnings Analysis.....	195
9.5.8 Partitioning Analysis.....	195

9.6	Software Testing.....	195
9.6.1	Purpose of Software Testing	196
9.6.2	Overview of DO-178C's Software Testing Guidance.....	198
9.6.2.1	Requirements-Based Test Methods	198
9.6.2.2	Normal and Robustness Tests	199
9.6.3	Survey of Testing Strategies	200
9.6.3.1	Equivalence Class Partitioning	201
9.6.3.2	Boundary Value Testing.....	202
9.6.3.3	State Transition Testing	203
9.6.3.4	Decision Table Testing	203
9.6.3.5	Integration Testing	203
9.6.3.6	Performance Testing	204
9.6.3.7	Other Strategies	204
9.6.3.8	Complexity Measurements.....	205
9.6.3.9	Summary and Characteristics of a Good Test.....	205
9.6.4	Test Planning	206
9.6.5	Test Development.....	207
9.6.5.1	Test Cases	207
9.6.5.2	Test Procedures.....	208
9.6.5.3	DO-178C Requirements.....	208
9.6.5.4	Low-Level Requirements Testing versus Unit Testing.....	209
9.6.5.5	Handling Requirements That Cannot Be Tested.....	209
9.6.5.6	Obtaining Credit for Multiple Levels of Testing	210
9.6.5.7	Testing Additional Levels of Requirements.....	210
9.6.6	Test Execution.....	210
9.6.6.1	Performing Dry Runs	211
9.6.6.2	Reviewing Test Cases and Procedures.....	211
9.6.6.3	Using Target Computer versus Emulator or Simulator.....	211
9.6.6.4	Documenting the Verification Environment.....	211
9.6.6.5	Test Readiness Review.....	212
9.6.6.6	Running Tests for Certification Credit.....	212
9.6.7	Test Reporting	213
9.6.8	Test Traceability.....	213
9.6.9	Regression Testing	214
9.6.10	Testability	215
9.6.11	Automation in the Verification Processes	215
9.7	Verification of Verification	216
9.7.1	Review of Test Procedures.....	217
9.7.2	Review of Test Results.....	218

9.7.3	Requirements Coverage Analysis	218
9.7.4	Structural Coverage Analysis.....	219
9.7.4.1	Statement Coverage (DO-178C Table A-7 Objective 7).....	220
9.7.4.2	Decision Coverage (DO-178C Table A-7 Objective 6).....	221
9.7.4.3	Modified Condition/Decision Coverage (DO-178C Table A-7 Objective 5).....	221
9.7.4.4	Additional Code Verification (DO-178C Table A-7 Objective 9)	222
9.7.4.5	Data Coupling and Control Coupling Analyses (DO-178C Table A-7 Objective 8).....	223
9.7.4.6	Addressing Structural Coverage Gaps.....	227
9.7.4.7	Final Thoughts on Structural Coverage Analysis.....	228
9.8	Problem Reporting.....	228
9.9	Recommendations for the Verification Processes	232
	References	236
	Recommended Readings	237
10.	Software Configuration Management	239
	Acronyms	239
10.1	Introduction.....	239
10.1.1	What Is Software Configuration Management?	239
10.1.2	Why Is Software Configuration Management Needed?.....	240
10.1.3	Who Is Responsible for Implementing Software Configuration Management?	242
10.1.4	What Does Software Configuration Management Involve?	242
10.2	SCM Activities.....	243
10.2.1	Configuration Identification	243
10.2.2	Baselines	244
10.2.3	Traceability	244
10.2.4	Problem Reporting	245
10.2.4.1	Problem Report Management with Multiple Stakeholders	245
10.2.4.2	Managing Open/Deferred Problem Reports... ..	248
10.2.5	Change Control and Review.....	249
10.2.6	Configuration Status Accounting	250
10.2.7	Release.....	251
10.2.8	Archival and Retrieval.....	252
10.2.9	Data Control Categories	253
10.2.10	Load Control	253
10.2.11	Software Life Cycle Environment Control	255

10.3	Special SCM Skills	256
10.4	SCM Data	256
10.4.1	SCM Plan.....	256
10.4.2	Problem Reports.....	257
10.4.3	Software Life Cycle Environment Configuration Index.....	257
10.4.4	Software Configuration Index	257
10.4.5	SCM Records	258
10.5	SCM Pitfalls	258
10.6	Change Impact Analysis.....	261
	References	265
11.	Software Quality Assurance	267
	Acronyms	267
11.1	Introduction: Software Quality and Software Quality Assurance (SQA)	267
11.1.1	Defining Software Quality	267
11.1.2	Characteristics of High-Quality Software	268
11.1.3	Software Quality Assurance	269
11.1.4	Examples of Common Quality Process and Product Issues.....	271
11.2	Characteristics of Effective and Ineffective SQA.....	271
11.2.1	Effective SQA.....	271
11.2.2	Ineffective SQA.....	273
11.3	SQA Activities.....	274
	References	278
12.	Certification Liaison	281
	Acronyms	281
12.1	What Is Certification Liaison?	282
12.2	Communicating with the Certification Authorities.....	283
12.2.1	Best Practices for Coordinating with Certification Authorities	284
12.3	Software Accomplishment Summary	287
12.4	Stage of Involvement (SOI) Audits.....	289
12.4.1	Overview of SOI Audits.....	289
12.4.2	Overview of the Software Job Aid	289
12.4.3	Using the Software Job Aid	293
12.4.4	General Recommendations for the Auditor	293
12.4.5	General Recommendations for the Auditee (the Applicant/Developer).....	300
12.4.6	SOI Review Specifics	303
12.4.6.1	SOI 1 Entry Criteria, Expectations, and Preparation Recommendations	303

12.4.6.2	SOI 2 Entry Criteria, Expectations, and Preparation Recommendations	305
12.4.6.3	SOI 3 Entry Criteria, Expectations, and Preparation Recommendations	308
12.4.6.4	SOI 4 Entry Criteria, Expectations, and Preparation Recommendations	311
12.5	Software Maturity Prior to Certification Flight Tests	313
	References	314

Part IV Tool Qualification and DO-178C Supplements

13. DO-330 and Software Tool Qualification.....	317
Acronyms	317
13.1 Introduction	317
13.2 Determining Tool Qualification Need and Level (DO-178C Section 12.2)	320
13.3 Qualifying a Tool (DO-330 Overview)	323
13.3.1 Need for DO-330	323
13.3.2 DO-330 Tool Qualification Process	325
13.4 Special Tool Qualification Topics.....	334
13.4.1 FAA Order 8110.49	334
13.4.2 Tool Determinism	334
13.4.3 Additional Tool Qualification Considerations.....	337
13.4.4 Tool Qualification Pitfalls	338
13.4.5 DO-330 and DO-178C Supplements	340
13.4.6 Using DO-330 for Other Domains	340
References	341
14. DO-331 and Model-Based Development and Verification.....	343
Acronyms	343
14.1 Introduction	343
14.2 Potential Benefits of Model-Based Development and Verification	345
14.3 Potential Risks of Model-Based Development and Verification	348
14.4 Overview of DO-331	351
14.5 Certification Authorities Recognition of DO-331	357
References	358
15. DO-332 and Object-Oriented Technology and Related Techniques.....	359
Acronyms	359
15.1 Introduction to Object-Oriented Technology	359
15.2 Use of OOT in Aviation.....	360

15.3 OOT in Aviation Handbook 361

15.4 FAA-Sponsored Research on OOT and Structural Coverage..... 362

15.5 DO-332 Overview 362

 15.5.1 Planning 363

 15.5.2 Development..... 363

 15.5.3 Verification..... 363

 15.5.4 Vulnerabilities 364

 15.5.5 Type Safety..... 364

 15.5.6 Related Techniques..... 365

 15.5.7 Frequently Asked Questions..... 365

15.6 OOT Recommendations..... 366

15.7 Conclusion..... 366

References 367

Recommended Readings 367

16. DO-333 and Formal Methods..... 369

 Acronyms 369

16.1 Introduction to Formal Methods 369

16.2 What Are Formal Methods? 371

16.3 Potential Benefits of Formal Methods 373

16.4 Challenges of Formal Methods..... 374

16.5 DO-333 Overview 376

 16.5.1 Purpose of DO-333 376

 16.5.2 DO-333 and DO-178C Compared 376

 16.5.2.1 Planning and Development..... 376

 16.5.2.2 Configuration Management, Quality Assurance, and Certification Liaison 377

 16.5.2.3 Verification 377

16.6 Other Resources 379

References 380

Part V Special Topics

17. Noncovered Code (Dead, Extraneous, and Deactivated Code) 383

 Acronyms 383

17.1 Introduction 383

17.2 Extraneous and Dead Code..... 383

 17.2.1 Avoiding Late Discoveries of Extraneous and Dead Code..... 385

 17.2.2 Evaluating Extraneous or Dead Code..... 386

17.3 Deactivated Code 388

 17.3.1 Planning..... 390

17.3.2	Development.....	391
17.3.3	Verification.....	392
	Reference.....	393
18.	Field-Loadable Software	395
	Acronyms.....	395
18.1	Introduction.....	395
18.2	What Is Field-Loadable Software?	396
18.3	Benefits of Field-Loadable Software.....	396
18.4	Challenges of Field-Loadable Software	397
18.5	Developing and Loading Field-Loadable Software	397
18.5.1	Developing the System to Be Field-Loadable.....	398
18.5.2	Developing the Field-Loadable Software	398
18.5.3	Loading the Field-Loadable Software.....	399
18.5.4	Modifying the Field-Loadable Software	400
18.6	Summary.....	401
	References	401
19.	User-Modifiable Software	403
	Acronyms.....	403
19.1	Introduction.....	403
19.2	What Is User-Modifiable Software?.....	403
19.3	Examples of UMS.....	405
19.4	Designing the System for UMS.....	405
19.5	Modifying and Maintaining UMS	408
	References	410
20.	Real-Time Operating Systems	411
	Acronyms.....	411
20.1	Introduction.....	412
20.2	What Is an RTOS?	412
20.3	Why Use an RTOS?	413
20.4	RTOS Kernel and Its Supporting Software	414
20.4.1	RTOS Kernel	415
20.4.2	Application Program Interface	415
20.4.3	Board Support Package.....	416
20.4.4	Device Driver.....	416
20.4.5	Support Libraries	418
20.5	Characteristics of an RTOS Used in Safety-Critical Systems.....	418
20.5.1	Deterministic.....	418
20.5.2	Reliable Performance.....	418
20.5.3	Compatible with the Hardware.....	419
20.5.4	Compatible with the Environment.....	419
20.5.5	Fault Tolerant.....	419

20.5.6	Health Monitoring.....	419
20.5.7	Certifiable.....	420
20.5.8	Maintainable.....	420
20.5.9	Reusable	421
20.6	Features of an RTOS Used in Safety-Critical Systems.....	421
20.6.1	Multitasking	421
20.6.2	Guaranteed and Deterministic Schedulability.....	421
	20.6.2.1 Scheduling between Partitions	422
	20.6.2.2 Scheduling within Partitions.....	422
20.6.3	Deterministic Intertask Communication	424
20.6.4	Reliable Memory Management.....	425
20.6.5	Interrupt Processing.....	425
20.6.6	Hook Functions	426
20.6.7	Robustness Checking	426
20.6.8	File System	426
20.6.9	Robust Partitioning.....	427
20.7	RTOS Issues to Consider	427
20.7.1	Technical Issues to Consider	427
	20.7.1.1 Resource Contention.....	427
	20.7.1.2 Priority Inversion	428
	20.7.1.3 Memory Leaks.....	429
	20.7.1.4 Memory Fragmentation	429
	20.7.1.5 Intertask Interference	429
	20.7.1.6 Jitter.....	429
	20.7.1.7 Vulnerabilities	429
20.7.2	Certification Issues to Consider	430
	20.7.2.1 Creating a Safe Subset.....	431
	20.7.2.2 User's Manual.....	431
	20.7.2.3 Reverse Engineering.....	431
	20.7.2.4 Deactivated Features	431
	20.7.2.5 Complexity	431
	20.7.2.6 Disconnect with the System	432
	20.7.2.7 Code Compliance Issues	432
	20.7.2.8 Error Handling Issues	432
	20.7.2.9 Problem Reporting.....	432
	20.7.2.10 Partitioning Analysis.....	433
	20.7.2.11 Other Supporting Software	433
	20.7.2.12 Target Testing	433
	20.7.2.13 Modifications.....	433
20.8	Other RTOS-Related Topics.....	434
20.8.1	ARINC 653 Overview	434
20.8.2	Tool Support	437
20.8.3	Open Source RTOSs	437
20.8.4	Multicore Processors, Virtualization, and Hypervisors.....	438

20.8.5 Security.....	439
20.8.6 RTOS Selection Questions	439
References	439
21. Software Partitioning	443
Acronyms	443
21.1 Introduction to Partitioning	443
21.1.1 Partitioning: A Subset of Protection.....	444
21.1.2 DO-178C and Partitioning	445
21.1.3 Robust Partitioning.....	446
21.2 Shared Memory (Spatial Partitioning).....	448
21.3 Shared Central Processing Unit (Temporal Partitioning)	449
21.4 Shared Input/Output.....	450
21.5 Some Partitioning-Related Challenges	451
21.5.1 Direct Memory Access	451
21.5.2 Cache Memory	451
21.5.3 Interrupts	452
21.5.4 Interpartition Communication	453
21.6 Recommendations for Partitioning	453
References	459
22. Configuration Data.....	461
Acronyms	461
22.1 Introduction.....	461
22.2 Terminology and Examples.....	462
22.3 Summary of DO-178C Guidance on Parameter Data	464
22.4 Recommendations.....	465
References	470
23. Aeronautical Data	471
Acronyms	471
23.1 Introduction	471
23.2 DO-200A: Standards for Processing Aeronautical Data.....	472
23.3 FAA Advisory Circular 20-153A	476
23.4 Tools Used for Processing Aeronautical Data.....	478
23.5 Other Industry Documents Related to Aeronautical Data	479
23.5.1 DO-201A: Standards for Aeronautical Information.....	479
23.5.2 DO-236B: Minimum Aviation System Performance Standards: Required Navigation Performance for Area Navigation	479
23.5.3 DO-272C: User Requirements for Aerodrome Mapping Information.....	480
23.5.4 DO-276A: User Requirements for Terrain and Obstacle Data	480

23.5.5	DO-291B: Interchange Standards for Terrain, Obstacle, and Aerodrome Mapping Data	480
23.5.6	ARINC 424: Standard, Navigation System Database	480
23.5.7	ARINC 816-1: Embedded Interchange Format for Airport Mapping Database	481
	References	482
24.	Software Reuse	485
	Acronyms	485
24.1	Introduction	485
24.2	Designing Reusable Components.....	487
24.3	Reusing Previously Developed Software	492
24.3.1	Evaluating PDS for Use in Civil Aviation Products.....	493
24.3.2	Reusing PDS That Was Not Developed Using DO-178[]	497
24.3.3	Additional Thoughts on COTS Software	498
24.4	Product Service History	502
24.4.1	Definition of Product Service History	502
24.4.2	Difficulties in Seeking Credit Using Product Service History	503
24.4.3	Factors to Consider When Claiming Credit Using Product Service History	504
	References	505
25.	Reverse Engineering	507
	Acronyms	507
25.1	What Is Reverse Engineering?	508
25.2	Examples of Reverse Engineering	509
25.3	Issues to Be Addressed When Reverse Engineering	509
25.4	Recommendations for Reverse Engineering	511
	References	518
26.	Outsourcing and Offshoring Software Life Cycle Activities	519
	Acronyms	519
26.1	Introduction	519
26.2	Reasons for Outsourcing	521
26.3	Challenges and Risks in Outsourcing	522
26.4	Recommendations to Overcome the Challenges and Risks	526
26.5	Summary	536
	References	536

Appendix A: Example Transition Criteria..... 537

Appendix B: Real-Time Operating System Areas of Concern 549

**Appendix C: Questions to Consider When Selecting a Real-Time
Operating System for a Safety-Critical System 555**

Appendix D: Software Service History Questions..... 561

Index 567