

Contents

- 1 Introduction..... 1**
 - 1.1 What Is Software?..... 1
 - 1.2 What Is Software Engineering? 29
 - 1.3 The Major Activities/Tasks to Be Performed
in Software Engineering 31
 - 1.4 The Popular Lifecycle/Process Models with the Existing
Software Engineering Paradigm 32
 - 1.4.1 The Waterfall Model 32
 - 1.4.2 The Incremental Development Models 34
 - 1.4.3 The Iterative Models 36
 - 1.4.4 More Popular Process Models 39
 - 1.4.5 General Comments to All Process Models Existing
with the Old-Established Software Engineering Paradigm..... 43
 - 1.5 Why the Current Software Is Not Sufficiently Engineered
at This Time to Fulfill the Role of “Foundation”..... 45
 - 1.6 What Does a Revolution Mean? 47
 - 1.6.1 Three Phases of Scientific Revolutions..... 47
 - 1.6.2 Progress Through Revolutions 48
 - 1.7 What Is NSE? 48
 - 1.8 Summary..... 57
 - 1.9 Points and Questions to Ponder 58
 - 1.10 Further Reading and Information Source 58
 - References..... 58
- 2 Is the Old-Established Software Engineering
Paradigm Entirely Out of Date?..... 61**
 - 2.1 The 20 Famous Software Disasters Reported..... 65
 - 2.1.1 Very High Project Failure Rate Reported..... 67
 - 2.2 What Is the Root Cause for Software Disasters
and Very High Software Project Failure Rate? 67
 - 2.3 The “Software” Definition Is Outdated 69

2.4	The Current Software Development Process	
	Models Are Out of Date	70
2.5	Current Software Development Methodologies Are Out of Date	71
2.6	The Existing Software Modeling Approaches Are Outdated	72
2.7	Current Software Testing Paradigm Is Out of Date	72
2.8	Current Software Quality Assurance Paradigm Is Out of Date	72
2.9	Current Software Visualization Paradigm Is Out of Date.....	73
2.10	Current Software Documentation Paradigm Is out of Date	73
2.11	Current Software Maintenance Paradigm Is Out of Date	73
2.12	Current Software Project Management Paradigm Is Out of Date	74
2.13	“The Mythical Man-Month” Is an Outcome of Linear Thinking; The “No Silver Bullet” Conclusion Is Out of Date	74
2.14	Summary	76
2.15	Points and Questions to Ponder	77
2.16	Further Reading and Information Source	77
	References.....	77
3	Foundation for Establishing NSE: Complexity Science	79
3.1	The Basis of Complexity Science.....	79
3.1.1	Linear and Nonlinear.....	80
3.1.2	Reductionism.....	80
3.1.3	Chaos Theory	80
3.1.4	System	81
3.1.5	System Categories	81
3.1.6	Linear System.....	81
3.1.7	Nonlinear System and Complex System.....	81
3.1.8	Feedback	82
3.1.9	Fractal.....	82
3.1.10	Fractal Dimension	82
3.1.11	Dynamical System	82
3.1.12	Dissipation Structure	82
3.1.13	Li–Yorke Theorem: Period Three Theorem	83
3.1.14	Self-Organization	83
3.1.15	Synergetics	83
3.1.16	Catastrophe Theory	83
3.1.17	Complex Adaptive System.....	84
3.1.18	Meta-Synthesis.....	84
3.1.19	Cellular Automata	84
3.1.20	Genetic Algorithm.....	85
3.1.21	Soliton	86
3.2	Linear Thinking and Nonlinear Thinking.....	86
3.3	The Essential Principles of Complexity Science	87
3.4	Applications of Complexity Science	88

3.5	Complexity Science and NSE.....	89
3.6	Summary.....	89
3.7	Points and Questions to Ponder	89
3.8	Further Reading and Information Source	89
	References.....	90
4	Prediction and Practices: A New Round of Industrial Revolution Driven by Complexity Science and a General Paradigm-Shift Framework	91
4.1	Prediction: A New Round of Industrial Revolution Driven by Complexity Science Is Coming	91
4.2	The Contribution and Limitation of Hall’s Systems Engineering Framework	92
4.3	The Background for the Innovation of FDS	93
4.4	The Objectives of Innovating FDS	93
4.5	The Description of FDS.....	94
4.5.1	The “Principles of Complexity Science” Axis.....	94
4.5.2	The “Environment” Axis.....	96
4.5.3	The “People/Logic” Axis	96
4.5.4	The “New Paradigm” Axis Modified from the “Knowledge/Skills” Axis in Hall’s Framework.....	97
4.5.5	The “Phases” (Workflows) Axis	97
4.6	The Major Features of FDS	98
4.7	Applications of FDS	99
4.8	Bringing Feedback to the Research and Development of Complexity Science.....	100
4.9	Summary.....	101
4.10	Points and Questions to Ponder	101
4.11	Further Reading and Information Source	101
	References.....	101
5	Outline of the NSE Paradigm	103
5.1	A Tree Will Not Fall at One Blow: The Difficulty in Software Engineering Revolution.....	103
5.2	The Objectives for Establishing NSE	105
5.3	The Strategy to Achieve the Objectives of NSE.....	106
5.4	The Establishment of NSE.....	106
5.5	The Structure of NSE.....	107
5.6	The Components of NSE.....	107
5.7	The Major Feature and Characteristics of NSE.....	109
5.8	Summary.....	112
5.9	Points and Questions to Ponder	112
5.10	Further Reading and Information Source	112
	References.....	113

6 The Techniques Innovated to Support NSE	115
6.1 Definitions	115
6.2 Holistic, Virtual, and Traceable Diagram Generation Technique.....	117
6.3 Virtual and Traceable Documentation Technique.....	119
6.4 Holistic and Intelligent Version Comparison Technique	121
6.5 Holistic and Dynamic Traceability Technique	122
6.6 Comprehensive Software Testing Technique Mainly Based on the Transparent-Box Method	122
6.7 Defect Prevention Driven Quality Assurance Technique	123
6.8 Test Case Efficiency Analysis and Test Case Minimization Technique.....	125
6.9 Refactoring Technique with Defect Prevention.....	126
6.10 Holistic MC/DC Test Coverage Analysis and Graphical Representation Technique.....	127
6.11 Assisted Test Case Design Technique	128
6.12 Intelligent Regression Test Case Selection Technique	128
6.13 Holistic, Actor–Action and Event–Response Driven, Traceable, Visual, and Executable Technique for Requirement Development.....	130
6.14 Synthesis Design and Incremental Growing Up (Implementation and Integration) Technique	131
6.15 Holistic, Global, and Side-Effect-Prevention Based Software Maintenance Technique.....	133
6.16 Summary.....	133
6.17 Points and Questions to Ponder.....	134
6.18 Further Reading and Information Source	134
References.....	134
7 NSE Software Engineering Visualization Paradigm	135
7.1 The Old-Established Software Engineering Visualization Paradigm Is Outdated	135
7.2 The Revolutionary Solution Offered by NSE.....	137
7.3 The 3J graphics (J-Chart, J-Diagram, and J-Flow).....	138
7.4 J-Chart.....	138
7.5 J-Diagram	140
7.6 J-Flow	148
7.7 Entire Software Life Cycle Visualization with NSE	153
7.8 Rich Options for Generating 3J Graphics.....	155
7.8.1 For J-Chart Generation.....	155
7.8.2 For J-Diagram and J-Flow Generation.....	160
7.9 The Major Features of NSE Software Visualization Paradigm	160
7.10 Applications	180
7.11 Self-Documenting.....	191

7.12	Summary	195
7.13	Points and Questions to Ponder	196
7.14	Further Reading and Information Source	196
	References	197
8	NSE Process Model	199
8.1	Some Experts' Expectations	199
8.2	All of the Existing Software Engineering Process Models Are Outdated	201
8.3	Outline of the Revolutionary Solution Offered with NSE	202
8.4	The Driving Forces and The Support Techniques	203
8.5	The Graphical Representation of the NSE Process Model	204
8.5.1	The Objectives of the Preprocess	206
8.5.2	The Objectives of the Main Process	207
8.5.3	The Objective of the Support Facility for Automated and Bidirectional Traceability	208
8.6	The Major Steps of the Preprocess	208
8.7	The Major Steps of the Main Process	213
8.8	The Support Facility for Automated and Bidirectional Traceability	224
8.9	The Manifestation of the Essential Principles of Complexity Science in the NSE Process Model	225
8.10	The Major Features and Characteristics of the NSE Process Model	226
8.11	Summary	234
8.12	Points and Questions to Ponder	235
8.13	Further Reading and Information Source	236
	References	236
9	The Facility for Automated and Self-Maintainable Traceability	237
9.1	The Importance of Requirement Traceability	238
9.2	The Problems Addressed	238
9.3	The Solution Offered with NSE	239
9.3.1	Part 1	240
9.3.2	Part 2	240
9.4	How It Works	242
9.4.1	Bidirectional Traceability Between the Test Cases and the Source Code Modules or Branches	245
9.4.2	Extending the Bidirectional Traceability to Include All Related Documents	246
9.5	The Major Features	249
9.5.1	Automated	249
9.5.2	Self-Maintainable	250

9.5.3 Methodology-Independent	250
9.5.4 Nonlinear, Bidirectional, and Parallel	250
9.5.5 Accurate	250
9.5.6 Precise	251
9.5.7 Extended to Include Software Project Management Documents.....	251
9.5.8 Extended to Include Web Pages	251
9.5.9 Extended for Multiproject Support	251
9.5.10 Dynamic	252
9.5.11 Easy to Add on at Any Time, In Any Status	253
9.6 Application	254
9.7 Summary.....	254
9.8 Points and Questions to Ponder	255
9.9 Further Reading and Information Source	256
References.....	256
10 NSE Software Development Methodology Driven by Defect Prevention and Traceability	257
10.1 Almost All Existing Software Development Methodologies Are Outdated.....	257
10.2 Outline of the Revolutionary Solution Offered by NSE	259
10.3 The Driving Forces for the Innovation of the NSE Software Development Methodology.....	263
10.4 The Related NSE Software Engineering Process Model	265
10.5 Graphical Presentation of the NSE Software Development Methodology	267
10.6 Application.....	270
10.6.1 Some Suggestions About the Applications of the NSE Software Development Methodology	270
10.7 The Major Features of the NSE Software Development Methodology.....	271
10.8 Summary	271
10.9 Points and Questions to Ponder	272
10.10 Further Reading and Information Source	272
References.....	272
11 Requirement Engineering Under NSE: Source Code Driven Dynamic Software Modeling.....	273
11.1 Are All the Existing Software Modeling Approaches Outdated?.....	273
11.2 Outline of the Revolutionary Solution Offered by NSE	276
11.3 Description of the HAETVE Technique	279
11.4 Applications of HAETVE.....	286
11.5 How to Make a Hard Copy of a Graphical Requirement Document	303

11.6	Suggestions for the Requirement Documentation Design	304
11.7	The Major Features of HAETVE.....	306
11.8	More About Dynamic Modeling.....	309
11.9	Summary	311
11.10	Points and Questions to Ponder	311
11.11	Further Reading and Information Source	311
	References.....	312
12	Design Engineering Under NSE	313
12.1	The Major Problem Addressed	313
12.2	Outline of the Solution for Software Design with NSE.....	314
12.3	Description of the Innovated “Synthesis Design and Incremental Growing Up” Technique	315
12.3.1	Basic Ideas	315
12.3.2	What is Synthesis? What is Analysis?	316
12.3.3	Recommendation for Graphic Document Creation/ Generation.....	318
12.3.4	Self-Documenting	320
12.3.5	Detailed System Hierarchy Design	321
12.3.6	Static Defect Prevention and Defect Propagation Prevention Through Traceability	321
12.3.7	Dynamic Defect Prevention and Defect Propagation Prevention	321
12.3.8	Data Structure Design	323
12.3.9	Detailed Logic Design of the Modules	323
12.4	Application.....	325
12.5	The Major Features of the Software Synthesis Design Technique	336
12.6	Summary	337
12.7	Points and Questions to Ponder	337
12.8	Further Reading and Information Source	338
	References.....	338
13	Coding Engineering with NSE.....	339
13.1	The Problems Addressed	339
13.2	The Solution: Software Coding Engineering with NSE Using the Synthesis Design and Incremental Integration Technique	341
13.3	Unit Testing and Integration Testing Support	349
13.4	MC/DC Test Coverage Measurement Support	353
13.4.1	Conclusion	361
13.5	Semiautomated Inspection Support	362
13.6	Defect Prevention Driven Quality Assurance in Programming	364
13.7	Quality Measurement for an Entire Software Product and Each of Its Components	366
13.8	Application.....	367

13.9	The Major Features	368
13.10	Summary	368
13.11	Points and Questions to Ponder	368
13.12	Further Reading and Information Source	369
	References	369
14	The Basis of Software Testing	371
14.1	The Purpose of Software Testing	371
14.2	Functional Testing and the Black-Box Method	373
14.3	Structural Testing and the White-Box Method	373
14.3.1	Test Coverage Metrics	374
14.3.2	Instrumentation Methods	374
14.4	Gray-Box Testing	375
14.5	Performance Testing and the Testing Method	376
14.6	Other Nonfunctional Testing	377
14.7	Unit Testing, Integration Testing, and System Testing	378
14.8	Regression Test After Code Modification	378
14.9	Object-Oriented Software Testing	378
14.10	Web Application Testing	380
14.11	Embedded Software Testing	381
14.12	GUI Operation Capture and Playback	382
14.13	Acceptance Testing	383
14.14	Why Should Software Testing Tools Be Used	383
14.15	The Major Drawback of the Major Existing Software Testing Paradigm and the Solution	383
14.16	Summary	384
14.17	Points and Questions to Ponder	384
14.18	Further Reading and Information Source	384
	References	384
15	Software Test Case Design	387
15.1	What Is a Test Case?	387
15.2	The Basis of Test Case Design	388
15.2.1	Equivalence Class Partition and Boundary Value Analysis	388
15.2.2	State Transition Analysis	389
15.2.3	Conditions Combination Method	389
15.3	Semiautomated Test Case Design	390
15.4	Test Case Efficiency Measurement	391
15.5	Test Case Minimization	391
15.6	NSE Test Case Design with HAETVE Technique for Both Functional Testing and Structural Testing	396
15.7	Automated Test Case Selection with Automated Test Case Execution	405
15.8	Summary	406

15.9	Points and Questions to Ponder	407
15.10	Further Reading and Information Source	407
	References.....	407
16	The NSE Software Testing Paradigm Based on the Transparent-Box Method	409
16.1	The Major Existing Software Testing Methods, Techniques, and Tools Are Outdated	409
16.2	The Transparent-Box Testing Method	411
16.3	The New Software Testing Paradigm Based on the Transparent-Box Testing Method.....	413
16.4	The Major Features of the New Software Testing Paradigm	417
16.5	A General Comparison Between the New Software Testing Paradigm and the Old One	429
16.6	Summary	432
16.7	Points and Questions to Ponder	432
16.8	Further Reading and Information Source	432
	References.....	432
17	NSE Software Quality Assurance Paradigm Driven by Defect Prevention.....	433
17.1	The Old-Established Software Quality Assurance Paradigm Is Outdated	433
17.2	Outline of NSE Software Quality Assurance Paradigm (NSE-SQA).....	435
17.3	Description of NSE Software Quality Assurance Paradigm.....	436
17.3.1	The Foundation of NSE-SQA	436
17.3.2	The Framework for Establishing NSE-SQA.....	436
17.3.3	The Purpose of NSE-SQA	437
17.3.4	Definitions.....	437
17.3.5	The Quality Assurance Strategy of NSE-SQA	439
17.3.6	The Implementation of the Quality Assurance Strategy of NSE-SQA	439
17.4	Application of NSE-SQA	460
17.5	The Major Features of NSE-SQA.....	460
17.6	Summary	463
17.7	Points and Questions to Ponder	464
17.8	Further Reading and Information Source	464
	References.....	464
18	NSE Software Maintenance Paradigm: Systematic, Disciplined, and Quantifiable	467
18.1	The Existing Software Maintenance Engineering Paradigm Is Outdated.....	467
18.2	Outline of the NSE Software Maintenance Paradigm	470

18.3	Description of NSE Software Maintenance Engineering Paradigm.....	476
18.4	Application	477
18.5	The Major Features.....	485
18.6	Summary	487
18.7	Points and Questions to Ponder	487
18.8	Further Reading and Information Source	487
	References.....	488
19	NSE Documentation Paradigm: Virtual, Traceable, and Consistent with the Source Code.....	489
19.1	The Old-Established Software Documentation Paradigm Is Outdated.....	489
19.2	Outline of NSE Documentation Paradigm.....	491
19.3	Description of the NSE Documentation Paradigm	494
19.3.1	The Critical Issues with the Old-Established Software Documentation Paradigm	494
19.3.2	The Solution Offered with NSE.....	495
19.3.3	The Objectives of the NSE Documentation Paradigm.....	496
19.3.4	Working with Dummy Programming.....	497
19.3.5	Working with NSE Software Visualization Paradigm	497
19.3.6	Working with HAETVE Requirement Development Technique	497
19.3.7	How It Works	500
19.3.8	Making a Software Product Visible in Multiple-Views	500
19.4	The Major Features of NSE Documentation Paradigm.....	505
19.5	Application	510
19.6	Summary	510
19.7	Points and Questions to Ponder	512
19.8	Further Reading and Information Source	514
	References.....	515
20	NSE Project Management Paradigm: Seamlessly Combined with the Project Development Process	517
20.1	The Old-Established Software Project Management Paradigm Is Outdated.....	517
20.2	Outline of the NSE Project Management Paradigm	518
20.3	The Foundation of NSE Project Management Paradigm.....	519
20.4	The Strategy of NSE Project Management Paradigm.....	520
20.5	People Oriented.....	521
20.6	Focusing on Maintenance	522
20.7	More Method and Tool Support.....	523
20.8	Combination of Product Development and Project Management.....	524
20.9	Finding Problems Early and Solving the Problems in Time.....	528

20.10	Quality Management.....	528
20.11	Multiple-Project Management	528
20.12	Summary	528
20.13	Points and Questions to Ponder	529
20.14	Further Reading and Information Source	530
	References.....	530
21	Algorithms Innovated for Establishing NSE	531
21.1	The Algorithm for Realizing Modified Condition/Decision Coverage Test Coverage Measurement.....	532
21.1.1	The Requirements	532
21.1.2	The Basic Idea.....	532
21.1.3	The Major Steps	533
21.1.4	Application.....	533
21.2	The Algorithm for Test Case Efficiency Analysis and Test Case Minimization.....	533
21.2.1	The Requirements	533
21.2.2	The Basic Idea.....	534
21.2.3	The Major Steps	535
21.2.4	Application.....	536
21.3	The Algorithm for Performance Analysis.....	536
21.3.1	The Requirements	536
21.3.2	The Basic Idea.....	537
21.3.3	The Major Steps	538
21.3.4	Application.....	539
21.4	The Algorithm for Cyclomatic Complexity Analysis.....	540
21.4.1	The Requirement.....	540
21.4.2	The Basic Idea.....	540
21.4.3	The Major Steps	541
21.4.4	Application.....	541
21.5	The Algorithm for Tracing the Execution Path of a Runtime Error	542
21.6	The Algorithm for the Layout of the Call Graph of a Program Using J-Chart Notations.....	543
21.7	The Algorithm for Holistic Version Comparison of a Software Product.....	543
21.8	The Algorithm for Memory Leak and Usage Violation Analysis	543
21.9	The Algorithm for Realizing the Traceability of the Diagrammed Source Code	546
21.10	The Algorithm for Dynamic Traceability	549
21.11	Summary	553
21.12	Points and Questions to Ponder	553
21.13	Further Reading and Information Source	555
	References.....	555

- 22 NSE Support Tools and NSE Support Platforms.....557**
 - 22.1 Full Software Development Lifecycle Support.....557
 - 22.2 The Product Development History557
 - 22.2.1 The First Generation: Hindsight557
 - 22.2.2 Second Generation: Panorama558
 - 22.2.3 Panorama++560
 - 22.3 Automated Tools Integrated with Panorama++560
 - 22.4 Panorama++ Product Installation.....560
 - 22.5 A Guided Tour of Panorama++ for C/C++565
 - 22.6 Network Floating License Support574
 - 22.7 The Major Features of Panorama++575
 - 22.8 Applications575
 - 22.9 Summary575
 - 22.10 Points and Questions to Ponder575
 - 22.11 Further Reading and Information Source575
 - References.....576

- 23 NSE Applications577**
 - 23.1 The Whole and Its Components: A General Comparison Between NSE and Other Approaches577
 - 23.2 What Makes NSE Special?579
 - 23.3 Applications in New Software Development.....579
 - 23.3.1 Benefits579
 - 23.3.2 Recommended Process580
 - 23.4 Applications in a Software Product Being Developed Using Other Approaches.....583
 - 23.5 Possible Combination with UML583
 - 23.5.1 About the Future of UML583
 - 23.5.2 Question to the Future of UML583
 - 23.5.3 Possible Combination with UML (NSE-UML?)584
 - 23.5.4 Possible Combination with CMMI (NSE-CMMI?).....585
 - 23.6 Possible Combination with Agile Software Development Approaches.....587
 - 23.6.1 Possible Combination with XP (NSE-XP?).....592
 - 23.7 Possible Combination with RUP (NSE-RUP?).....593
 - 23.8 Support for CBSE593
 - 23.9 Summary.....594
 - 23.10 Points and Questions to Ponder595
 - 23.11 Further Reading and Information Source595
 - References.....595

- 24 Candidates of “Silver Bullet”597**
 - 24.1 Is “The Mythical Man-Month” an Outcome of Linear Thinking, Reductionism, and Superposition Principle?597
 - 24.1.1 A Great book.....598
 - 24.1.2 Limitation.....599

- 24.2 Is the “No Silver Bullet” Conclusion Outdated? 599
- 24.3 The First Candidate of “Silver Bullet” 602
- 24.4 The Second Candidate of “Silver Bullet” 604
- 24.5 Can the “Silver Bullet” Defined by Brooks Slay
the “Werewolves” Defined by Him? 605
- 24.6 What Kind of “Silver Bullet” Can be Used to Slay
the “Werewolves” Defined by Brooks? 607
- 24.7 The Third Candidate of “Silver Bullet”:
The Entire NSE Paradigm 609
 - 24.7.1 What Is NSE: The Whole and Its Components 610
 - 24.7.2 The Components of NSE 614
 - 24.7.3 The Major Features and Characteristics of NSE 616
 - 24.7.4 The Major Differences Between NSE and the
Old-Established Software Engineering Paradigm 618
 - 24.7.5 Qualification as a Candidate of “Silver Bullet”
for Slaying Software “Werewolves” 628
- 24.8 Summary 647
- 24.9 Points and Questions to Ponder 648
- 24.10 Further Reading and Information Source 649
- References 649

- Appendix A: Software Requirements Specification Template
To Be Used with NSE 651**

- Appendix B: An Example About How to Realize 100% MC/DC
(Modified Condition/Decision Coverage) for a Program Unit 675**

- Appendix C: How to Control/Simulate the Return Values
of a Program Unit Being Tested 699**

- Appendix D: Hints for Answering the “Points
and Questions to Ponder” in Each Chapter 703**

- Glossary 727**

- Index 731**

New Software Engineering Paradigm Based on
Complexity Science

An Introduction to NSE

Xiong, J.

2011, XXXV, 746 p. With online files/update., Hardcover

ISBN: 978-1-4419-7325-2