

SOFTWARE ENGINEERING FOUNDATIONS

**A SOFTWARE SCIENCE
PERSPECTIVE**

The CRC Series in Software Engineering



Series Editor-in Chief: Prof. Yingxu Wang, PhD

- Vol. 1 Y. Wang and G. King (2000)
Software Engineering Processes: Principles and Applications
- Vol. 2 Y. Wang (2007)
Software Engineering Foundations: A Software Science Perspective
- Vol. 3 Y. Wang (2008)
Denotational Mathematics: Rigorous Means for Software Science and
Cognitive Informatics
- Vol. 4 Y. Wang (2009)
Software Engineering Notations: Seamlessly Refining Systems to
Architectures, Behaviors, and Code
- Vol. 5 Y. Wang (2009)
Software Engineering Measurement and Analysis: Metrics for Quantitative
Software Engineering

SOFTWARE ENGINEERING FOUNDATIONS

A SOFTWARE SCIENCE PERSPECTIVE

YINGXU WANG

PROFESSOR, PhD, P.ENG, F.WIF, SMIEEE, MACM
University of Calgary



CRC Press

Boca Raton London New York Washington, D.C.

Part I Fundamental Principles of Software Engineering x

Library of Congress Cataloging-in-Publication Data

Wang, Yingxu

Software Engineering Foundations: A Software Science Perspective / Yingxu Wang

p. cm

Includes bibliographical references and index.

ISBN 0-8493-1931-5

1. Software engineering. I. Wang, Y. II. Title.

© 2007 by CRC Press

ISBN 978-0-8493-1931-0

To my parents, wife, and daughters

Great knowledge sees all in one. Small knowledge
breaks down into the many.

Chuang Tzu (399 – 295 BC)

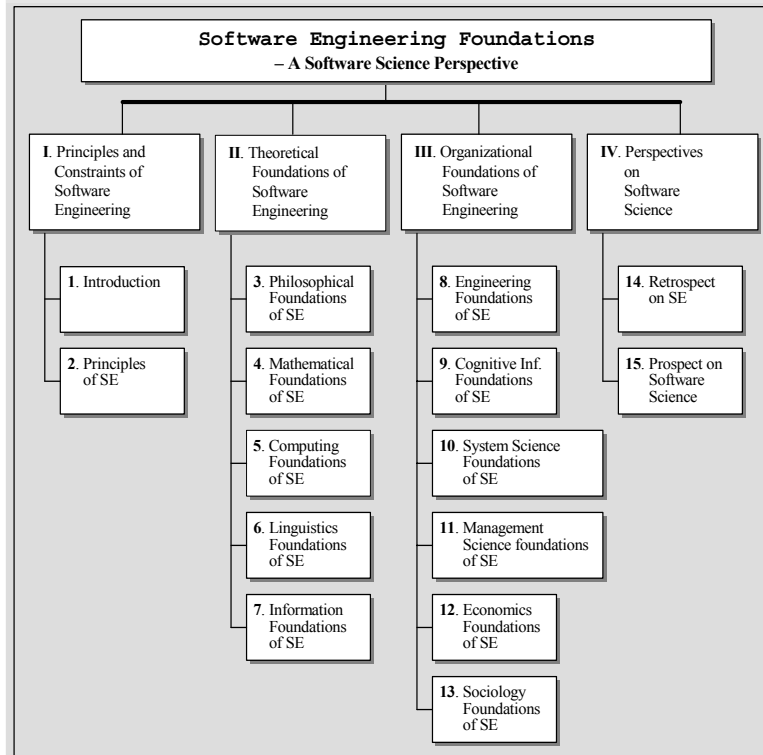
Problems that are created by our current level of thinking
cannot be solved by that same level of thinking.

Albert Einstein (1879 – 1955)

The more science becomes divided into specialized disciplines, the more
important it becomes to find unifying principles.

Herman Haken (1977)

Summary of Contents



Summary of Contents

Software Engineering Foundations

A Software Science Perspective

Part I. Principles and Constraints of Software Engineering

1. Introduction
2. Principles of Software Engineering

Part II. Theoretical Foundations of Software Engineering

3. Philosophical Foundations of Software Engineering
4. Mathematical Foundations of Software Engineering
5. Computing Foundations of Software Engineering
6. Linguistics Foundations of Software Engineering
7. Information Science Foundations of Software Engineering

Part III. Organizational Foundations of Software Engineering

8. Engineering Foundations of Software Engineering
9. Cognitive Informatics Foundations of Software Engineering
10. System Science Foundations of Software Engineering
11. Management Science Foundations of Software Engineering
12. Economics Foundations of Software Engineering
13. Sociology Foundations of Software Engineering

Part IV. Perspectives on Software Science

14. Retrospect on Software Engineering
15. Prospect on Software Science

Bibliography

Appendixes

- A. Mathematical Symbols, Notations, and Abbreviations
- B. Constraints of Software Engineering
- C. Empirical Principles of Software Engineering
- D. Models of Entities and Structures of Software Engineering
- E. Wang's Laws of Software Engineering
- F. Wang's Formal Principles of Software Engineering
- G. The Type System of Software Engineering
- H. Meta Processes of Software Engineering
- I. Algebraic Process Relations of Software Engineering
- J. Deductive Semantics of Software Engineering
- K. Formal Models of the ATM System in RTPA
- L. List of Figures
- M. List of Tables

Index

Table of Contents

Summary of Contents	vii
Preface	xxxv
Acknowledgments	xliv
About the Author	vlvii
Part I Principles and Constraints of Software Engineering	1
1 Introduction	5
1.1 Overview	7
1.1.1 Software Engineering: Status and Problems	10
1.1.2 Myths on Software Engineering	12
1.2 Characteristics of Software Engineering	14
1.2.1 Perceptions on Software	15
1.2.1.1 The Mathematical Metaphor of Software	16
1.2.1.2 The Product Metaphor of Software	16
1.2.1.3 The Informatics Metaphor of Software	17
1.2.2 Perceptions on Software Engineering	18
1.2.3 Software Engineering as an Engineering Discipline	21
1.2.4 Hierarchy of Abstraction and Descriptivity in Software Engineering	23
1.2.4.1 The Hierarchical Abstraction Model of System Descriptivity (HAMSD)	24
1.2.4.2 Software Engineering Practice: Can Microtech be Used to Denote Nanotech?	26
1.3 Basic Constraints of Software Engineering	28
1.3.1 The Software Engineering Constraint Model	28
1.3.2 Cognitive Constraints of Software Engineering	29
1.3.2.1 Intangibility	30
1.3.2.2 Complexity	30
1.3.2.3 Indeterminacy	31

1.3.2.4 Diversity	32
1.3.2.5 Polymorphism	32
1.3.2.6 Inexpressiveness	33
1.3.2.7 Inexplicit Embodiment	34
1.3.2.8 Unquantifiable Quality Measures	35
1.3.3 Organizational Constraints of Software Engineering	35
1.3.3.1 Time dependency	36
1.3.3.2 Conservative Productivity	36
1.3.3.3 Labor-Time Interlock	37
1.3.4 Resources Constraints of Software Engineering	38
1.3.4.1 Costs	38
1.3.4.2 Human Dependency	39
1.3.4.3 Hardware Dependency	39
1.4 Approaches to Software Engineering	40
1.4.1 Programming Methodologies	41
1.4.2 Software Development Models	41
1.4.3 Automated Software Engineering	42
1.4.4 Formal Methods	43
1.4.5 Software Engineering Processes	43
1.4.6 Theoretical Foundations of Software Engineering	44
1.5 Transdisciplinary Foundations of Software Engineering	45
1.5.1 Philosophical Foundations	45
1.5.2 Mathematical Foundations	46
1.5.3 Computing Foundations	46
1.5.4 Linguistics Foundations	47
1.5.5 Information Science Foundations	47
1.5.6 Engineering Foundations	48
1.5.7 Cognitive Informatics Foundations	48
1.5.8 Systems Science Foundations	48
1.5.9 Management Science Foundations	49
1.5.10 Economics Foundations	49
1.5.11 Sociology Foundations	50
1.6 The Architecture of this Book	50
1.7 Summary	55
Questions and Research Opportunities	62
2 Principles of Software Engineering	67
2.1 Introduction	69
2.2 Pioneer Pursuits of Principles for Software Engineering	70
2.2.1 Parnas' Principles of Software Engineering	71
2.2.1.1 Information Hiding	71
2.2.1.2 Modularization	72
2.2.1.3 Engineering Approach	72

xvi Table of Contents

2.2.1.4 Professional Responsibility	73
2.2.1.5 Documentation	73
2.2.2 Hoare's Principles of Software Engineering	74
2.2.2.1 Professionalism	74
2.2.2.2 Vigilance	75
2.2.2.3 Sound Theoretical Knowledge	75
2.2.2.4 Using Tools	75
2.2.2.5 Abstraction	75
2.2.2.6 Structured Programming	76
2.2.2.7 Readability	76
2.2.3 Brooks' Principles of Software Engineering	76
2.2.3.1 Complexity	77
2.2.3.2 Conformity	78
2.2.3.3 Changeability	78
2.2.3.4 Invisibility	78
2.2.4 Wasserman's Principles of Software Engineering	79
2.2.4.1 Abstraction	79
2.2.4.2 Methods and Notations	80
2.2.4.3 Prototyping	80
2.2.4.4 Modularity and Architecture	80
2.2.4.5 Lifecycle and Process	81
2.2.4.6 Reuse	81
2.2.4.7 Metrics	81
2.2.4.8 Tools and Integrated Environments	81
2.2.5 IEEE SESC's Principles of Software Engineering	82
2.2.6 IEEE Software Magazine's Principles of Software Engineering	83
2.2.6.1 Reviews and Inspections	84
2.2.6.2 Information Hiding	84
2.2.6.3 Incremental Development	84
2.2.6.4 User Involvement	85
2.2.6.5 Automated Revision Control	85
2.2.6.6 Internet Development	85
2.2.6.7 Programming Languages Hall of Fame	85
2.2.6.8 Capacity Maturity Model	86
2.2.6.9 Object-Oriented Programming	86
2.2.6.10 Component-Based Development	86
2.2.6.11 Metrics and Measurement	87
2.3 A Unified Framework of Software Engineering Principles	87
2.3.1 Elicitation of Fundamental Principles of Software Engineering	88
2.3.2 The Unified Framework of Software Engineering Principles	90
2.3.3 Description of the Fundamental Principles of Software	90

Engineering	
2.3.3.1 Abstraction	90
2.3.3.2 Decomposition/Modularization	92
2.3.3.3 Information Hiding	93
2.3.3.4 Engineering Approach	93
2.3.3.5 Professionalism	94
2.3.3.6 Tools and Environments	94
2.3.3.7 Documentation	95
2.3.3.8 Stepwise Refinement	96
2.3.3.9 Prototyping	96
2.3.3.10 Adopting Engineering Notations	97
2.3.3.11 Process Modeling	98
2.3.3.12 Reuse	98
2.3.3.13 Measurements and Metrics	99
2.3.3.14 Cognitive Complexity Control	100
2.3.3.15 Formal Requirement Specification	101
2.3.3.16 Systematic Quality Assurance	101
2.3.3.17 Review and Inspection	102
2.3.3.18 Management Engineering	102
2.3.3.19 Acquiring Domain Knowledge	103
2.3.3.20 Customer Involvement	104
2.3.3.21 Feasibility Analysis	104
2.3.3.22 Improving Comprehensibility	105
2.3.3.23 Exception Handling	106
2.3.3.24 Divide and Conquer	106
2.3.3.25 Explicit Embodiment	107
2.3.3.26 Establishing Theoretical Foundations	108
2.3.3.27 Architecture and Behavior Modeling	108
2.3.3.28 Standardization	109
2.3.3.29 Systems Engineering	110
2.3.3.30 Engineering Organization	110
2.3.3.31 Cognitive Engineering	110
2.4 Software Engineering Principles as Measures to its Constraints	111
2.4.1 Principles for Coping with the Cognitive Constraints	111
2.4.2 Principles for Coping with the Organizational Constraints	114
2.4.3 Principles for Coping with the Resource Constraints	115
2.4.4 A Systematic View on Mapping between the Principles and Constraints	116
2.5 Summary	118
Questions and Research Opportunities	125

Part II Theoretical Foundations of Software Engineering	129
3 Philosophical Foundations of Software Engineering	133
3.1 Introduction	135
3.2 Philosophy of Sciences and Engineering	136
3.2.1 The Natural World and the Abstract World	137
3.2.2 The Basic Axioms about Nature	138
3.2.3 Epistemology and Foundationalism	139
3.2.4 Holism vs. Reductionism	141
3.2.5 Positivism vs. Rationalism	142
3.2.6 Empiricism and Objectivity	143
3.2.7 Determinism vs. Indeterminism	145
3.2.8 Natural Intelligence vs. Artificial Intelligence	145
3.2.9 Ethical Philosophies of Engineering	147
3.3 Formal Inference Methodologies	148
3.3.1 Logical Argumentations	148
3.3.2 Deductive Inferences	150
3.3.3 Inductive Inferences	151
3.3.4 Abductive Inferences	153
3.3.5 Analogical Inferences	154
3.4 The Nature of Software	155
3.4.1 The Three Situations where Software is Needed	155
3.4.2 The Behavioral Space of Software	156
3.4.3 Properties of Software	157
3.4.3.1 The Cognitive Properties of Software	158
3.4.3.2 The Intelligent Behavioral Properties of Software	158
3.4.3.3 The System Properties of Software	160
3.5 Philosophy of Software Engineering	160
3.5.1 The Cognitive Characteristics of Software Engineering	161
3.5.1.1 The Abstraction and Intangibility of Software	161
3.5.1.2 The Inherited Complexity and Diversity	161
3.5.1.3 The Changeability or Malleability of Software	161
3.5.1.4 The Difficulty of Establishing and Stabilizing Requirements	162
3.5.1.5 The Requirement of Varying Problem Domain Knowledge	162
3.5.1.6 The Indeterminacy and Polysolvability in Design	162
3.5.1.7 The Polyglotics and Polymorphism in Implementation	163
3.5.1.8 The Dependability of Interactions between Software, Hardware, and Humans	163
3.5.2 The Nature of Software Engineering	163

3.5.2.1 Programming: Virtualization vs. Realization	163
3.5.2.2 Problem Domains: Infinitive vs. Limited	164
3.5.2.3 Effort Distribution: Design Intensive vs. Repetitive Production	164
3.5.2.4 Implementation: Specificity vs. Generality	165
3.5.2.5 Universal Logical Description vs. Domain-Specific Description	166
3.5.2.6 Process Standardization vs. Product Standardization	166
3.5.3 Software Engineering Validation Methodologies	166
3.6 Murphy's Laws: The Practitioners' Philosophy for Software Engineering	167
3.6.1 Murphy's Laws on General Engineering	168
3.6.2 Murphy's Laws on Software Engineering	169
3.7 Summary	170
Questions and Research Opportunities	177
4 Mathematical Foundations of Software Engineering	181
4.1 Introduction	183
4.2 Set Theory	185
4.2.1 Sets and Properties	185
4.2.1.1 Set Notations and Terminologies	185
4.2.1.2 Set Operations	187
4.2.1.3 Algebraic Laws of Sets	189
4.2.2 Sequences and Ordered Sets	190
4.2.2.1 Pairs and Tuples	190
4.2.2.2 Sequences	191
4.2.2.3 Lists	191
4.2.2.4 Ordered Sets	192
4.2.3 Relations	192
4.2.3.1 Binary Relations	192
4.2.3.2 Compositions of Relations	193
4.2.3.3 Properties of Relations	194
4.2.3.4 Cumulative Relations of Programs	195
4.3 Algebra Systems	196
4.3.1 Abstraction in Algebra Systems	196
4.3.1.1 Abstract Algebra	196
4.3.1.2 Boolean Algebra	196
4.3.1.3 Process Algebra	197
4.3.1.4 Concept Algebra	197
4.3.1.5 System Algebra	198
4.3.2 Functions	198
4.3.2.1 Notations of Functions	198
4.3.2.2 Inverse Functions	199
4.3.2.3 Composition of Functions	199

xx Table of Contents

4.3.3 Algebraic Operations	200
4.4 Mathematical Logic	201
4.4.1 Propositional Logic	202
4.4.1.1 Propositions	202
4.4.1.2 Propositional Logic Operations	203
4.4.1.3 Laws of Propositional Algebra and Logical Inferences	204
4.4.2 Predicate Logic	205
4.4.2.1 Taxonomy of Predicates	206
4.4.2.2 Concept Construction with Predicate Logic	207
4.4.2.3 Inferences in Predicate Logic	208
4.5 Denotational Mathematics for Software Engineering	209
4.5.1 Fundamental Elements in Modeling Software Systems	210
4.5.2 The Need for Denotational Mathematics in Software Engineering	212
4.5.2.1 Problems Yet to be Solved	212
4.5.2.2 New Problems Require New Forms of Mathematics	212
4.5.3 The Big-R Notation	214
4.6 Real-Time Process Algebra (RTPA)	217
4.6.1 The Process Metaphor of Software Systems	217
4.6.1.1 Process Algebra	218
4.6.1.2 Real-Time Process Algebra	219
4.6.2 The Structure of RTPA	221
4.6.3 The Type System of RTPA	222
4.6.3.1 Primitive Types and the Type-Suffix Convention	222
4.6.3.2 Definitions of the Primitive Types of RTPA	222
4.6.3.3 Equivalence between Primitive Types	224
4.6.4 Meta Processes of RTPA	225
4.6.4.1 Structure of the RTPA Meta Processes	226
4.6.4.2 Formal Description of the RTPA Meta Processes	227
4.6.5 Process Relations and Algebraic Operations of RTPA	233
4.6.5.1 Structure of the RTPA Process Relations	233
4.6.5.2 Formal Description of the RTPA Process Relations	236
4.7 The RTPA Methodology for Software System Modeling and Refinement	241
4.7.1 The RTPA Methodology	242
4.7.2 System Architecture Modeling and Refinement in RTPA	245
4.7.2.1 The System Architecture	245
4.7.2.2 The CLM Schema	246
4.7.2.3 The CLM Objects	246
4.7.3 System Static Behavior Modeling and Refinement	248
4.7.3.1 System Static Behaviors	248

4.7.3.2 Process Schemas	248
4.7.3.3 Process Implementation	249
4.7.4 System Dynamic Behavior Modeling and Refinement	249
4.7.4.1 System Dynamic Behaviors	249
4.7.4.2 Dynamic Behaviors Deployment	250
4.7.4.3 Dynamic Behaviors Dispatch	251
4.8 RTPA: Notations for Software Engineering	252
4.8.1 Modeling Component-Level Problems using RTPA	252
4.8.1.1 Existing Approaches to ADT Specification	253
4.8.1.2 Architectural Specification in RTPA	254
4.8.1.3 Static Behavior Specification in RTPA	255
4.8.1.4 Dynamic Behavior Specification in RTPA	255
4.8.2 Modeling System-Level Problems using RTPA	256
4.8.2.1 The Conceptual Model of the ATM	257
4.8.2.2 Formal Description of the ATM Architectures	258
4.8.2.3 Formal Description of the ATM Static Behaviors	259
4.8.2.4 Formal Description of the ATM Dynamic Behaviors	260
4.9 Summary	263
Questions and Research Opportunities	271
5 Computing Foundations of Software Engineering	277
5.1 Introduction	279
5.2 Basic Computational Models	281
5.2.1 Basic Operations in Computing	281
5.2.2 Automata	284
5.2.2.1 Automata and Finite State Machines (FSMs)	284
5.2.2.2 Approaches to Describe FSMs	286
5.2.2.3 Description of Software Behaviors by FSMs	288
5.2.2.4 FSM Composition and Refinement	291
5.2.2.5 Deterministic and Nondeterministic Automata	292
5.2.2.6 Usage of Automata	293
5.2.3 Turing Machines	294
5.2.3.1 The Abstract Model of Computing	294
5.2.3.2 Formal Description of Turing Machines	295
5.2.3.3 The Nature of Computing	297
5.2.4 von Neumann Machines	298
5.2.4.1 The Stored-Program Concept	299
5.2.4.2 The von Neumann Architecture of Computers	299
5.2.5 Cognitive Machines	302
5.2.5.1 The Wang Architecture of Computers	302
5.2.5.2 Cognitive Computers	303
5.3 Data Object Modeling and Manipulation	304
5.3.1 Types and Data Structures	305

xxii Table of Contents

5.3.1.1	Type Systems of Programming Languages	305
5.3.1.2	Primitive Types	307
5.3.1.3	Derived and Advanced Types	310
5.3.1.4	System Architectural Types	312
5.3.2	Basic Data Modeling Techniques	314
5.3.2.1	Identifiers	314
5.3.2.2	Variables and Constants	317
5.3.2.3	Expressions	318
5.3.3	Formal Type Theory	319
5.3.3.1	Type Rules	319
5.3.3.2	Formal Type Systems	321
5.3.3.3	Complex Type Rules for the RTPA Derived Types	321
5.3.4	Abstract Data Types	324
5.3.4.1	The Generic Model of ADTs	325
5.3.4.2	Modeling Complex Data Structures and Component Architectures by ADTs	326
5.3.4.3	Typical ADTs Modeled in RTPA	327
5.4	Behavioral Modeling and Manipulation	331
5.4.1	Internal Behaviors Modeling	332
5.4.1.1	Basic Control Structures (BCS's)	333
5.4.1.2	Control Flow Graphs	333
5.4.2	Iterative and Recursive Behaviors Modeling	335
5.4.2.1	Formal Description of Iterations	336
5.4.2.2	Formal Description of Recursions	339
5.4.2.3	Comparative Analysis of Iterations and Recursions	342
5.4.3	External and Interactive Behaviors Modeling	345
5.4.3.1	Memory Manipulations	345
5.4.3.2	Events Handling	350
5.5	Program Modeling: Coordination of Computational Behaviors with Data Objects	351
5.5.1	The Unified Mathematical Model of Programs	352
5.5.1.1	The Abstract Model of Statements	352
5.5.1.2	The Abstract Model of Processes	353
5.5.1.3	The Abstract Model of Programs	353
5.5.2	Programs Modeling at Component Level	355
5.5.2.1	Algorithms	355
5.5.2.2	Classes and Object-Oriented	356
5.5.2.3	Patterns	361
5.5.3	Programs Modeling at System Level – Frameworks	369
5.6	Resources and Processes Modeling and Manipulation	373
5.6.1	Abstract Models of Computing Systems	373
5.6.2	Architectures of Operating Systems	376
5.6.2.1	The Generic Architecture of Operating Systems	376
5.6.2.2	The Unix™ and Linux™ Operating Systems	377

5.6.2.3 The Windows™ XP Operating Systems	378
5.6.3 Computing Resources Manipulation	379
5.6.3.1 Process Management	379
5.6.3.2 CPU Scheduling	380
5.6.3.3 Memory Management	381
5.6.3.4 File System Management	382
5.6.3.5 I/O System Management	382
5.6.3.6 Communication Management	384
5.6.3.7 Network Management	385
5.6.4 Real-Time/Embedded Resources and Processes Manipulation	386
5.6.4.1 The Architecture of RTOS+	387
5.6.4.2 The Task Scheduler of RTOS+	388
5.6.4.3 Process Dispatching of RTOS+	389
5.7 Summary	391
Questions and Research Opportunities	403
6 Linguistics Foundations of Software Engineering	411
6.1 Introduction	413
6.2 Fundamentals of Linguistics	415
6.2.1 Taxonomy of Linguistics	415
6.2.2 Syntaxes	416
6.2.3 Semantics	419
6.2.4 Grammars	420
6.2.4.1 Properties of Grammars	421
6.2.4.2 The Universal Grammar	422
6.2.4.3 The Deductive Grammar of English	422
6.3 Formal Language Theory	425
6.3.1 Alphabet	426
6.3.2 Strings	426
6.3.3 Expressions	428
6.3.4 Grammar Theories	429
6.3.4.1 Production Rules of Grammars	429
6.3.4.2 Taxonomy of Grammars	429
6.3.5 Languages	433
6.3.6 BNF and EBNF	435
6.4 Syntaxes of Programming Languages	437
6.4.1 Lexical Analyses	438
6.4.1.1 Taxonomy of Lexical Entities in Programming Languages	439
6.4.1.2 Lexical Analysis of Programs	440
6.4.2 Syntax Definitions and Descriptions	440
6.4.3 Syntactical Analyses	442
6.4.3.1 Basic Syntactical Analysis Techniques	442

xxiv Table of Contents

6.4.3.2 Description of Parsing Results by Syntax Trees	444
6.4.4 Syntactical Analyses of RTPA	445
6.4.4.1 Description of the RTPA Syntax in LL(k)	445
6.4.4.2 Description of Special RTPA Grammar Rules by Syntactic Predicates	446
6.4.4.3 Parsing RTPA Specifications	448
6.5 Semantics of Programming Languages	449
6.5.1 Taxonomy of Semantics	450
6.5.1.1 Target Semantics	450
6.5.1.2 Operational Semantics	451
6.5.1.3 Denotational Semantics	451
6.5.1.4 Axiomatic Semantics	451
6.5.1.5 Algebraic Semantics	452
6.5.1.6 Deductive Semantics	453
6.5.2 Denotational Semantics	453
6.5.2.1 Syntactic and Semantic Domains of Denotational Semantics	454
6.5.2.2 Description of Syntactic Domains of the Sample Language SPL	456
6.5.2.3 Semantic Analysis using Denotational Semantics	456
6.5.2.4 Semantics of Programs in SPL	461
6.5.3 Deductive Semantics	462
6.5.3.1 The Mathematical Model of Software Semantics	463
6.5.3.2 Deductive Semantics of Programs at Different Levels of Compositions	466
6.5.3.3 Properties of Software Semantics	470
6.6 Semantics of RTPA	472
6.6.1 Semantics of RTPA Meta Processes	472
6.6.1.1 The Assignment Process	473
6.6.1.2 The Evaluation Process	473
6.6.1.3 The Addressing Process	474
6.6.1.4 The Memory Allocation Process	475
6.6.1.5 The Memory Release Process	475
6.6.1.6 The Read Process	476
6.6.1.7 The Write Process	476
6.6.1.8 The Input Process	476
6.6.1.9 The Output Process	477
6.6.1.10 The Timing Process	477
6.6.1.11 The Duration Process	478
6.6.1.12 The Increase Process	478
6.6.1.13 The Decrease Process	479
6.6.1.14 The Exception Detection Process	479
6.6.1.15 The Skip Process	480
6.6.1.16 The Stop Process	481

6.6.2 Semantics of RTPA Process Relations	481
6.6.2.1 The Sequential Process Relation	482
6.6.2.2 The Jump Process Relation	484
6.6.2.3 The Branch Process Relation	485
6.6.2.4 The Switch Process Relation	486
6.6.2.5 The While-Loop Process Relation	488
6.6.2.6 The Repeat-Loop Process Relation	489
6.6.2.7 The For-Loop Process Relation	490
6.6.2.8 The Function Call Process Relation	491
6.6.2.9 The Recursive Process Relation	492
6.6.2.10 The Parallel Process Relation	493
6.6.2.11 The Concurrent Process Relation	494
6.6.2.12 The Interleave Process Relation	495
6.6.2.13 The Pipeline Process Relation	496
6.6.2.14 The Interrupt Process Relation	497
6.6.3 Semantics of System and System Process Dispatching	498
6.6.3.1 The System Process	498
6.6.3.2 The Time-Driven Dispatching Process Relation	499
6.6.3.3 The Event-Driven Dispatching Process Relation	500
6.6.3.4 The Interrupt-Driven Dispatching Process Relation	501
6.7 Linguistic Perspectives on Software Engineering	502
6.7.1 Comparative Analysis of Natural and Programming Language Theories	503
6.7.2 Principles of Programming Language Design	504
6.7.2.1 Abstraction and Complexity Control	504
6.7.2.2 Efficiency	504
6.7.2.3 Expressivity	505
6.7.2.4 Simplicity	505
6.7.2.5 Uniformity	506
6.7.2.6 Orthogonality	506
6.7.2.7 Comprehensibility and Readability	506
6.7.3 Characteristics of Programming Languages	506
6.7.3.1 Fundamental Requirements for Programming	506
6.7.3.2 Limitations of Programming Languages	507
6.8 Summary	508
Questions and Research Opportunities	522
7 Information Science Foundations of Software Engineering	527
7.1 Introduction	529
7.2 Classic Information Theory	530
7.2.1 Shannon's Perception on Information	531
7.2.2 The Physical Meaning of Classic Information	532
7.2.2.1 The Concept of Entropy	533
7.2.2.2 The Laws of Thermodynamics	533

xxvi Table of Contents

7.2.2.3 Transformation between Information Entropy and Thermal Entropy	535
7.2.3 Domain of Classical Information Theory	536
7.2.4 Subjectivity of Classic Information Theory	537
7.3 Contemporary Informatics	538
7.3.1 Information: The Third Essence of Nature	538
7.3.2 Measurement of Information	539
7.3.3 From Machine Informatics to Cognitive Informatics	540
7.3.3.1 Cognitive Informatics	541
7.3.3.2 Perspective on Information in Cognitive Informatics	542
7.3.3.3 The Role of Information in Mankind Evolution	543
7.4 Informatics Laws of Software	543
7.4.1 Equivalence between I-M-E	544
7.4.1.1 The Equivalence of Matter and Energy	544
7.4.1.2 Transformation between Matter, Energy, and Information	545
7.4.2 Informatics Laws and Properties of Software	547
7.4.2.1 Abstraction	548
7.4.2.2 Generality	548
7.4.2.3 Cumulativeness	549
7.4.2.4 Dependency on Cognition	549
7.4.2.5 Multi-Dimensional Behavioral Space	549
7.4.2.6 Sharability	550
7.4.2.7 Physically Dimensionless	550
7.4.2.8 Weightless	550
7.4.2.9 Transformability between I-M-E	550
7.4.2.10 Multiple Representation Forms	551
7.4.2.11 Multiple Carrying Media	551
7.4.2.12 Multiple Transmission Forms	551
7.4.2.13 Dependency on Media	552
7.4.2.14 Dependency on Energy	552
7.4.2.15 Wearless and Time Dependency	552
7.4.2.16 Conservation of Information Entropy and Thermal Entropy	552
7.4.2.17 Information-based Quality Attributes	552
7.4.2.18 Susceptible to Distortion	552
7.4.2.19 Scarcity	553
7.5 Information Theories for Software Engineering	554
7.5.1 The Informatics Metaphor of Software	554
7.5.2 Informatics Laws that Constrain Software Behaviors	555
7.5.3 The Informatics Attributes of Software Quality	556
7.6 Summary	557
Questions and Research Opportunities	564

Part III Organizational Foundations of Software Engineering	569
8 Engineering Foundations of Software Engineering	575
8.1 Introduction	577
8.2 Generic Engineering Approaches	578
8.2.1 Engineering: A Concept Emerged from the Industrial Revolutions	579
8.2.2 Science and the Generic Scientific Method	581
8.2.3 Engineering vs. Science	583
8.2.3.1 Science and Scientists	584
8.2.3.2 Engineering and Engineers	585
8.2.3.3 Relationship between Science and Engineering	586
8.2.4 Fundamental Goals and Constraints of Engineering	587
8.2.5 Generic Engineering Approaches	589
8.2.6 The Generic Engineering Maturity Model (EMM)	590
8.3 Basic Engineering Principles	593
8.3.1 Principles of Engineering Organization	593
8.3.2 Principles of Engineering Technology	594
8.3.3 Principles of Engineering Management	595
8.3.4 Principles of Engineering Professionalism	595
8.4 Engineering Principles for Software Engineering	596
8.4.1 The Engineering Characteristics of Software Engineering	596
8.4.2 Division of Labor	597
8.4.3 Characteristics of Software Engineering in the Engineering Age	599
8.4.4 Unique Principles of Software Engineering	600
8.4.5 Professionalism of Software Engineering	602
8.4.5.1 Professionalism of Software Engineers	602
8.4.5.2 Ethical Practice in Software Engineering	604
8.5 The Theory of Software Engineering Organization	606
8.5.1 Basic Properties of Coordinative Work in Engineering	606
8.5.1.1 The Mechanisms of Coordinative Workload and Effort	607
8.5.1.2 The Rate of Interpersonal Coordination	608
8.5.1.3 The Overhead of Interpersonal Coordination	610
8.5.1.4 The Nature of Coordinative Work in Engineering	612
8.5.2 Laws of Work Organization in Software Engineering	614
8.5.2.1 The Law of Incompressibility of Software Engineering Workload	614
8.5.2.2 The Laws of Interchangeability between Labor and Time in Software Engineering	615

xxviii Table of Contents

8.5.2.3 The Laws of the Shortest Duration of Coordinative Work in Software Engineering	616
8.5.3 The Mythical Man-Month Explained	621
8.5.4 Decision Optimization in Software Engineering	623
8.5.4.1 Optimization of Project Organization for the Shortest Duration	623
8.5.4.2 Optimization of Project Organization for the Lowest Effort/Cost	626
8.5.4.3 Optimization of Project Organization by Controlling the Interpersonal Coordination Rate	629
8.6 Empirical Software Engineering	631
8.6.1 Software Engineering Case Studies	631
8.6.2 Software Engineering Experiments	632
8.6.3 Software Engineering Trials	633
8.6.4 Software Engineering Benchmarking	635
8.6.4.1 The IBM European Benchmarks on Software Engineering Practices	636
8.6.4.2 The SEPRM Benchmarks on Software Engineering Processes	637
8.6.5 Software Engineering Standardization	639
8.6.5.1 Software Development Standards	639
8.6.5.2 Software Quality Standards	640
8.6.5.3 Software Engineering Process Standards	641
8.7 Summary	642
Questions and Research Opportunities	649
9 Cognitive Informatics Foundations of Software Engineering	655
9.1 Introduction	657
9.2 Cognitive Informatics	660
9.2.1 Cognitive Philosophy	660
9.2.2 Neural Informatics Foundations of the Brain	664
9.2.2.1 Neurons and Synapses	664
9.2.2.2 Physiological Structure of the Brain	666
9.2.2.3 Cognitive Models of Memories	668
9.2.3 The Emergence of Cognitive Informatics	674
9.2.4 The Theoretical Framework of Cognitive Informatics	676
9.2.4.1 The Fundamental Theories of Cognitive Informatics	677
9.2.4.2 The Domain of Cognitive Informatics	677
9.3 Cognitive Informatics Models of the Brain	679
9.3.1 The Layered Reference Model of the Brain (LRMB)	680
9.3.1.1 The Architecture of LRMB	680
9.3.1.2 The Functional Layers of LRMB	682
9.3.1.3 The Configuration of the Cognitive Processes of	685

	LRMB	
	9.3.2 Cognitive Properties of Internal Information	686
	9.3.3 Natural Intelligence vs. Artificial Intelligence	689
	9.3.3.1 The Nature of Intelligence	689
	9.3.3.2 Taxonomy of Intelligence	690
	9.3.3.3 The Model of Natural Intelligence	691
	9.3.3.4 Measurement of Intelligence	692
	9.3.3.5 Theory of Learning and Knowledge Acquisition	696
	9.3.4 The Cognitive Model of the Brain	697
9.4	Cognitive Informatics Models of Knowledge Representation	701
	9.4.1 The Hierarchical Neural Cluster (HNC) Model of Memory	702
	9.4.2 The Object-Attribute-Relation (OAR) Model of Internal Information Representation	702
	9.4.3 The Extended OAR Model of the Brain	706
	9.4.4 The Cognitive Mechanisms of Long-Term Memory	708
	9.4.4.1 Cognitive Properties of LTM	709
	9.4.4.2 When is memory created in LTM?	710
	9.4.4.3 How is memory created in LTM?	712
	9.4.5 The Memory Capacity of Human Brains	713
9.5	Cognitive Informatics for Software Engineering	715
	9.5.1 Cognitive Constraints on Software Productivity	716
	9.5.2 Software Engineering Psychology	717
	9.5.3 The Cognitive Foundation of Software Comprehension	719
	9.5.4 Software Engineering Skills and Experiences	722
	9.5.5 Software Agent Systems	724
9.6	Cognitive Complexity of Software	725
	9.6.1 The Relative Cognitive Weights of Generic Software Structures	726
	9.6.2 Psychological Experiments on the Cognitive Weights	728
	9.6.3 Calibration of the Relative Cognitive Weights of BCS's	729
9.7	Summary	730
	Questions and Research Opportunities	741
10	System Science Foundations of Software Engineering	747
	10.1 Introduction	749
	10.2 System Philosophies	750
	10.2.1 The System Metaphor for Modeling Complex Entities	751
	10.2.2 Holism	753
	10.2.3 Systematic Thinking	753
	10.3 Abstract Systems and System Topology	755
	10.3.1 Mathematical Models of Abstract Systems	755
	10.3.1.1 The Mathematical Model of Closed Systems	755

xxx Table of Contents

10.3.1.2 The Mathematical Model of Open Systems	757
10.3.2 Taxonomy of Systems	760
10.3.2.1 Concrete and Abstract Systems	760
10.3.2.2 Physical and Social Systems	762
10.3.2.3 Finite and Infinite Systems	762
10.3.2.4 Closed and Open Systems	764
10.3.2.5 Static and Dynamic Systems	764
10.3.2.6 Linear and Nonlinear Systems	765
10.3.2.7 Continuous and Discrete Systems	765
10.3.2.8 Precise and Fuzzy Systems	765
10.3.2.9 Determinate and Indeterminate Systems	766
10.3.2.10 White-Box and Black-Box Systems	766
10.3.2.11 Intelligent and Nonintelligent Systems	766
10.3.2.12 Maintainable and Nonmaintainable Systems	766
10.3.3 Magnitudes of Systems	767
10.3.3.1 System Sizes, Magnitudes, and Complexities	767
10.3.3.2 Taxonomy of System Magnitudes	769
10.3.4 Hierarchical Architectures of Systems	770
10.3.5 The System Organization Tree	772
10.3.6 System Cohesion and Coupling	774
10.3.6.1 The Border of Systems	774
10.3.6.2 System Cohesion and Coupling	775
10.4 System Algebra	776
10.4.1 Relational Operations of Systems	776
10.4.1.1 Algebraic Relations of Closed Systems	776
10.4.1.2 Algebraic Relations of Open Systems	777
10.4.1.3 Relations between Closed and Open Systems	779
10.4.2 Algebraic Operations of Systems	780
10.4.2.1 System Conjunction	780
10.4.2.2 System Difference	784
10.4.2.3 System Composition	786
10.4.2.4 System Decomposition	790
10.5 Principles of System Science	791
10.5.1 System Fusions	791
10.5.2 System Functions and Behaviors	793
10.5.3 Work Done by Systems	794
10.5.4 The Maximum Output of Systems	796
10.5.5 System Equilibrium and Organization	797
10.5.5.1 The Generic IPO Model of Systems	797
10.5.5.2 Laws of System Equilibrium and Organization	798
10.5.6 System Synchronization and Coordination	801
10.5.7 System Dissimilation	802
10.5.7.1 Dissimilation of Nonmaintainable Systems	802
10.5.7.2 Dissimilation of Maintainable Systems	804

10.6 Software System Engineering	805
10.6.1 The Abstract Model of Computing Systems	806
10.6.2 The Hierarchical Model of Software Systems	807
10.6.2.1 The Hierarchical Structure of Software Systems	807
10.6.2.2 The Hierarchical Structure of Software Engineering Processes and Work Products	807
10.6.3 The ISO/IEC 15288 System Engineering Model for Software Engineering	808
10.6.4 Software Engineering Phenomena as System Engineering Problems	810
10.7 The Complexity Theory of Software Systems	813
10.7.1 Computational Complexity	814
10.7.1.1 Taxonomy of Computational Problems	815
10.7.1.2 Time Complexity of Algorithms	816
10.7.1.3 Space Complexity of Algorithms	817
10.7.2 Symbolic and Control Flow Complexities	818
10.7.2.1 Symbolic Complexity of Software Systems	818
10.7.2.2 Control Flow Complexity of Software Systems	818
10.7.3 The Cognitive Complexities of Software Systems	820
10.7.3.1 The Operational Complexity of Software Systems	820
10.7.3.2 The Architectural Complexity of Software Systems	823
10.7.3.3 The Cognitive Complexity of Software Systems	825
10.7.4 Software System Complexity Analysis	827
10.7.4.1 Comparative Case Studies on the Complexity Models of Software Systems	827
10.7.4.2 The Symbolic vs. Cognitive Sizes of Software Systems	830
10.7.5 Cohesion and Coupling Complexities of Software Systems	831
10.7.5.1 Cohesion of Software Systems	832
10.7.5.2 Coupling of Software Systems	833
10.7.5.3 Comparative Analysis of Software System Cohesions and Couplings	833
10.8 Summary	835
Questions and Research Opportunities	849
11 Management Science Foundations of Software Engineering	855
11.1 Introduction	857
11.2 Principles of Management Science	859

xxxii Table of Contents

11.2.1	Classic Management Thought	860
11.2.2	Architecture of Management Science	861
11.2.2.1	Functions of Management	861
11.2.2.2	The System Model of Management	864
11.2.3	Fundamental Theory of Management Science	864
11.2.3.1	Why Management is Needed in Work Organization?	865
11.2.3.2	The First Principle of Management	867
11.2.3.3	Gains from Division of Labor	868
11.2.3.4	The Second Principle of Management	873
11.2.3.5	Wang's Work Organization Theory for Coordinative Work Management	874
11.3	Decision Theories	875
11.3.1	The Mathematical Model of Decision Making	876
11.3.1.1	The Principle of Choices	877
11.3.1.2	Decisions and Decision Making	879
11.3.1.3	Strategies and Criteria for Decision Making	879
11.3.1.4	The Structure of Rational Decision Making	880
11.3.2	Decision Making Processes	882
11.3.2.1	The Cognitive Process of Decision Making	882
11.3.2.2	Formal Description of the Decision Making Process	884
11.3.3	Static Decision Making Strategies	886
11.3.3.1	Decision Making under Certainty	888
11.3.3.2	Decision Making under Uncertainty	889
11.3.3.3	Decision Making under Risks	892
11.3.4	Game Theory	896
11.3.4.1	The Formal Model of Games	896
11.3.4.2	Properties of Games	899
11.3.4.3	Behaviors of Zero-Sum Games	901
11.3.4.4	Behaviors of Nonzero-Sum Games	905
11.3.5	Decision Grid Theory	907
11.3.5.1	The Formal Model of Decision Grids	908
11.3.5.2	Serial Decision Grids with Unlimited Trials	909
11.3.5.3	Serial Decision Grids with Limited Trials	916
11.4	Quality Systems	918
11.4.1	Quality Principles	918
11.4.1.1	Attributes of Quality	918
11.4.1.2	Formal Models of Quality	920
11.4.2	Quality Control and Assurance	923
11.4.2.1	Quality Control Systems	923
11.4.2.2	Quality Assurance Techniques	925
11.4.3	Quality Management Systems	927
11.4.3.1	Total Quality Management (TQM)	927

11.4.3.2	The ISO 9000 Quality System	928
11.4.3.3	The ISO 9126 Quality System	929
11.5	Software Engineering Management	932
11.5.1	Taxonomy of Software Engineering Management	932
11.5.2	The Software Engineering Process Reference Model (SEPRM)	935
11.5.2.1	The SEPRM Process Model	935
11.5.2.2	The SEPRM Capability Model	938
11.5.2.3	The SEPRM Capability Determination Methodology	940
11.6	Summary	943
	Questions and Research Opportunities	957
12	Economics Foundations of Software Engineering	963
12.1	Introduction	965
12.2	Fundamental Principles of Economics	967
12.2.1	Basic Axioms of Economics	967
12.2.1.1	Demand vs. Supply	967
12.2.1.2	The Principle of Resource Scarcity	968
12.2.1.3	The Law of Market Conservation	968
12.2.1.4	The Law of Maximizing Profits	969
12.2.2	Economic Equilibrium between Demands and Supplies	970
12.2.3	The Behaviors of Market Systems	971
12.2.3.1	Simple Modes of Economic Equilibriums	973
12.2.3.2	Complex Modes of Economic Equilibriums	977
12.2.3.3	The Adaptive Equilibrium Mechanisms of Market Systems	978
12.3	Economic Models	980
12.3.1	Production Models	980
12.3.2	Cost Models	981
12.3.3	Market Models	982
12.4	Dynamic Values of Money and Assets	983
12.4.1	Dynamics of Money	984
12.4.2	Dynamics of Asset's Values	985
12.4.3	Cumulative Values of Cash Flows	986
12.4.3.1	The Uniform Payment Series	986
12.4.3.2	The Linear Gradient Payment Series	987
12.4.3.3	The Geometric Gradient Payment Series	987
12.5	Economic Analyses	988
12.5.1	Project Cost Analyses	989
12.5.2	Project Benefit-Cost Analyses	990
12.5.3	Project Payback Period Analyses	992
12.5.4	Project Rate of Return Analyses	993
12.6	Software Engineering Economics	995

xxxiv Table of Contents

12.6.1	Elements of Software Engineering Costs	995
12.6.1.1	Analysis of Software Engineering Costs	995
12.6.1.2	Analysis of Software Engineering Revenues	997
12.6.2	Software Engineering Project Costs Estimation using FEMSEC	997
12.6.2.1	The FEMSEC Model of Software Engineering Costs	997
12.6.2.2	The FEMSEC Method for Software Engineering Project Costs Determination	999
12.6.3	Software Engineering Project Costs Estimation using COCOMO	1005
12.6.3.1	The Conceptual Model of COCOMO	1005
12.6.3.2	The Basic COCOMO Model	1006
12.6.3.3	The Intermediate COCOMO Model	1007
12.6.3.4	The Detailed COCOMO Model	1007
12.6.3.5	The COCOMO II Model	1008
12.6.4	Economic Analyses of Software Projects	1009
12.6.4.1	Estimations of Costs and Revenues of Software Projects	1009
12.6.4.2	Cumulated Value of Operating Costs	1011
12.6.4.3	Cumulated Present Value of Revenues	1011
12.6.4.4	Annual and Cumulated Depreciations of Equipment	1011
12.6.4.5	Project Benefit-Cost Ratios	1012
12.6.4.6	Project Payback Periods	1012
12.6.4.7	Project Rate of Return	1014
12.6.5	The Software Legacy Cost Model	1014
12.6.5.1	Development Costs vs. Maintenance Costs	1014
12.6.5.2	The Software Legacy Maintenance Cost Model	1015
12.7	Summary	1017
	Questions and Research Opportunities	1028

13 Sociology Foundations of Software Engineering 1033

13.1	Introduction	1035
13.2	Principles of Sociology	1036
13.2.1	Social Structures	1037
13.2.1.1	Individuals	1037
13.2.1.2	Groups	1038
13.2.1.3	Organizations	1038
13.2.1.4	Sectors	1039
13.2.1.5	Societies	1040
13.2.2	Social Behaviors	1040
13.2.2.1	Social Functions and Relations	1040

Table of Contents **xxxv**

13.2.2.2 Social Roles	1041
13.2.2.3 Social Systems	1043
13.2.3 Social Norms	1043
13.2.3.1 Cultures	1043
13.2.3.2 Values	1044
13.2.3.3 Socialization	1045
13.2.3.4 The Social Philosophy of Confucianism	1045
13.3 Social Psychology	1046
13.3.1 The Fundamental Human Traits	1046
13.3.1.1 Axiomatic Human Traits	1047
13.3.1.2 The Hierarchical Model of Basic Human Needs	1048
13.3.2 Human Perceptions and Behaviors	1050
13.3.2.1 Emotions	1051
13.3.2.2 Motivations	1053
13.3.2.3 Attitudes	1055
13.3.2.4 The Motivation/Attitude-Driven Behavioral Model	1056
13.3.3 Collective Behaviors	1058
13.3.3.1 Social Conformity	1058
13.3.3.2 Social Synchronization	1059
13.3.3.3 Coactions	1059
13.3.3.4 Coordination	1060
13.3.3.5 Groupthink	1060
13.3.3.6 Social Dilemmas	1061
13.3.3.7 Social Loafing	1061
13.4 Theory of Social Organization	1063
13.4.1 Classical Thought of Social Organization	1063
13.4.1.1 Principles of Social Organization	1063
13.4.1.2 Classic Models of Social Organization	1064
13.4.2 The Formal Model of Social Organization	1066
13.4.2.1 The Formal Organization Tree	1067
13.4.2.2 Formal Models of Social Organization	1069
13.4.2.3 Coordinative Work Organization	1072
13.4.3 The Formal Model of Socialization	1074
13.5 Sociology and Software Engineering	1077
13.5.1 Social Organization of Software Engineering	1077
13.5.1.1 The Role of the Information Economy in Postindustrial Societies	1078
13.5.1.2 Maximizing Strengths of Individual Motivations in Software Engineering	1078
13.5.1.3 Social Environments of Software Engineering	1079
13.5.1.4 Ergonomics for Software Engineering	1080
13.5.2 Theory for Large-Scale Software Engineering Project Organization	1081

xxxvi Table of Contents

13.5.3 The Human Factors in Software Engineering	1085
13.5.3.1 Taxonomy of Human Factors	1085
13.5.3.2 Types of Human Errors	1086
13.5.3.3 The Mathematical Model of Human Errors	1087
13.5.3.4 The Random Properties of Human Errors	1089
13.5.3.5 The Theoretical Foundation of Quality Assurance in Creative Work	1090
13.6 Summary	1092
Questions and Research Opportunities	1106

Part IV Perspectives on Software Science 1111

14 Retrospect on Software Engineering 1115

14.1 Introduction	1117
14.2 Infrastructures of Software Engineering	1118
14.2.1 The Process Infrastructure of Software Engineering	1119
14.2.2 Process-Based Software Engineering (PBSE)	1121
14.2.2.1 The Organizational Model of PBSE	1122
14.2.2.2 Software Engineering Process System Establishment	1124
14.2.2.3 Software Engineering Process System Assessment	1128
14.2.2.4 Software Engineering Process System Improvement	1131
14.3 Software Industry Organization	1134
14.3.1 The Nature of the Software Industry	1135
14.3.2 Principles of Software Industry Organization	1137
14.3.2.1 Basic Principles of Software Industrial Organization	1137
14.3.2.2 Separation of Software Designers, Builders, Quality Assurors, and Maintainers in Software Engineering	1138
14.3.2.3 Distributed Time-Shared Development in Software Engineering	1139
14.3.3 A Perspective on the Software Maintenance Crisis	1140
14.3.3.1 The Mathematical Model of Software Maintenance Crisis	1140
14.3.3.2 Reasons Behind Software Maintenance Crises	1141
14.3.3.3 Solutions to Software Maintenance Crisis	1142
14.4 Essential Knowledge towards Excellent Software Engineers	1143
14.4.1 Basic Constraints of Software Engineering	1145

14.4.2 Empirical Principles of Software Engineering	1146
14.4.3 Laws of Software Engineering	1149
14.4.4 Formal Principles of Software Engineering	1157
14.5 Impact of the Theoretical Foundations on Software Engineering	1165
14.5.1 The Cognitive Principles of Knowledge Engineering	1166
14.5.1.1 The Effort Model of Knowledge Creation and Acquisition	1166
14.5.1.2 The Complexity Model of Knowledge Creation	1168
14.5.1.3 The Cognitive Model of Knowledge Spaces of Multidisciplinary Knowledge	1169
14.5.2 Expected Impacts of Wang's Laws and Theorems to Software Engineering	1171
14.5.3 Students' Feedback	1175
14.6 Summary	1179
Questions and Research Opportunities	1187
15 Prospect on Software Science	1191
15.1 Introduction	1193
15.2 The Formal Knowledge Systems	1194
15.2.1 The Framework of Formal Knowledge	1194
15.2.2 The Roles of Formal and Empirical Knowledge	1197
15.3 A Discipline of Software Science	1199
15.3.1 Software Science: Software Engineering in the 21st Century	1200
15.3.2 Architecture of Software Science	1201
15.3.3 Denotational Mathematics for Software Science	1202
15.3.3.1 Concept Algebra	1203
15.3.3.2 System Algebra	1206
15.3.3.3 RTPA	1206
15.4 Impacts of Software Science on Computing	1208
15.4.1 Autonomic Computing	1208
15.4.1.1 From Imperative Computing to Autonomic Computing	1210
15.4.1.2 Behaviorism Foundations of Autonomic Computing	1211
15.4.1.3 Cognitive Informatics Foundations of Autonomic Computing	1214
15.4.1.4 Denotational Mathematics Foundations of Autonomic Computing	1216
15.4.1.5 Intelligent Science Foundations of Autonomic Computing	1216
15.4.2 Intelligent Code Generation	1217
15.4.3 Hyper-Programming: New Facets of the Software	1218

xxxviii Table of Contents

Architectural Framework	
15.4.3.1 The Architecture of Hyper-Programming	1219
15.4.3.2 Syntactic Relations between RTPA, UML, and C++	1221
15.4.3.3 The Framework of the Hyper-Programming Environment	1224
15.4.3.4 Applications of the Hyper-Programming System	1226
15.5 Epilogue	1229
Bibliography	1231
Appendixes	1279
A. Mathematical Symbols, Notations, and Abbreviations	1279
B. Constraints of Software Engineering	1289
C. Empirical Principles of Software Engineering	1291
D. Models of Entities and Structures of Software Engineering	1295
E. Wang's Laws of Software Engineering	1323
F. Wang's Formal Principles of Software Engineering	1331
G. The Type System of Software Engineering	1339
H. Meta Processes of Software Engineering	1341
I. Algebraic Process Relations of Software Engineering	1343
J. Deductive Semantics of Software Engineering	1345
K. Formal Model of the ATM System in RTPA	1359
L. List of Figures	1377
M. List of Tables	1387
Index	1393