

# CONTENTS

<b>List of Figures</b>	<b>xv</b>
<b>Acknowledgments</b>	<b>xvii</b>
<b>Chapter 1 Introduction</b>	<b>1</b>
1.1 The Retention of Software Jobs	2
1.2 Depth of Experience	2
1.3 The Scope of This Book	3
1.4 The Nature of Computer Science	4
1.5 The Future of Computer Science	4
1.6 The Essence of Philosophy	5
1.7 Why Autonomy?	6
1.8 An Architecture for Autonomy	8
1.9 Other Notes	9
<b>Chapter 2 Prologue</b>	<b>11</b>
2.1 How This Book Originated	11
2.2 The Importance of Management	12
2.3 The Tie-in with Autonomy	13
2.4 Major Themes of This Book	14
2.5 The Challenge of a New Idea	14
2.6 The Importance of Visualization	15
2.7 The Move Toward Autonomy	16
2.8 Why I Wrote This Book	17
2.9 Merging Theory and Practice	18
2.10 The Pace of Computer Science	19
2.11 The Importance of Cognitive Dynamics	21
<b>Chapter 3 The Philosophical Foundations of Computer Software Design</b>	<b>23</b>
3.1 The Philosophical Origins of Computer Science	23
3.2 Influence of the Cognitive Philosophers	25
3.3 Abstracting the Human Thought System	27
3.4 The Philosophical Foundations of Software Development	28
3.5 The Phenomenon of Reality	29

- 3.6 The Phenomenon of Subjectivity 30
- 3.7 Low-Cost Software Development 31
- 3.8 “On Budget and On Schedule” 33
- 3.9 The Time to Completion: Schedule 34
- 3.10 Philosophy and Successful Design 35

**Chapter 4 The Philosophical Imperatives of Architectural Design 36**

- 4.1 The Manager as Architect 36
- 4.2 The Manager as Teacher 37
- 4.3 The Manager as Social Worker 38
- 4.4 The Manager as Axman 39
- 4.5 The Philosophical Imperatives of Architectural Design 39
- 4.6 Availability of the Manager 40
- 4.7 Project Manager: 10 Key Attributes and Responsibilities 40
- 4.8 Philosophical Aspects of Engineering 43
- 4.9 The Importance of Finishing the Job 44
- 4.10 Visualizing an Architecture 44
- 4.11 The Role of Intuition in Design 47
- 4.12 “Sufficient Reality” and Inference in the Design Process 48
- 4.13 Dialectics in the Achievement of Sufficient Reality 49
- 4.14 The Relationship of Logic to Software Architectures 50
- 4.15 The Logic of the Systems Design 53

**Chapter 5 Project and Task Organization 54**

- 5.1 The Role of Organization 55
- 5.2 The Ability to Organize 57
  - 5.2.1 Traditional Hierarchical Project Organization 57
- 5.3 The Difficulty of Communication 60
- 5.4 The Title of “Manager” 61
- 5.5 The Flat, Nonhierarchical Organization 61
- 5.6 Projects, Tasks, and Work Units 65
- 5.7 Large Organizations and Staffing 66
- 5.8 Staffing Up: The Initial Team 69
  - 5.8.1 The Initial Team 69
  - 5.8.2 Phase One Team Expansion 70
  - 5.8.3 Phase Two Team Expansion 72
- 5.9 Balancing Hardware and Software 73
- 5.10 Incremental Deliveries 75
- 5.11 Functional Organization 76
- 5.12 Interface Protocols of the Organization 77

5.13	Completion of the Task	77
5.14	Detecting the “Fraud”	78
<b>Chapter 6</b>	<b>The Philosophy of Communication</b>	<b>80</b>
6.1	“Sanity Is an Achievement!”	81
6.2	Gauging Understanding	82
6.3	Internal Team Communication Protocols	82
6.4	External Team Communication Protocols	84
6.5	Technical English as the Medium	85
6.6	Engineers as Technical Writers	87
6.7	Documentation: Articulation of the Requirements and Design	87
6.8	The SRD: Software Requirements Document	88
<b>Chapter 7</b>	<b>Software Management Standards</b>	<b>91</b>
7.1	Three Good Standards	91
7.1.1	JPL-STD-D-4000	92
7.1.2	MIL-STD-498	92
7.1.3	DOD-STD-2167A	92
7.2	Aspects Addressed by a Standard	93
7.3	Preparing to Select the Standard	94
7.4	Standards for Implementation	96
7.4.1	Waiving the Standard	97
<b>Chapter 8</b>	<b>The Estimation of Software Cost</b>	<b>98</b>
8.1	Sponsor Costing Issues	99
8.2	Types of Cost Estimates	101
8.3	“Lines of Code” Metrics	101
8.4	The Major Work Areas, Functions, and Tasks that Must be Included in the Estimation of Cost	103
8.5	The Detailed Cost Estimate	105
8.6	The SRD as a Contract	106
<b>Chapter 9</b>	<b>The Exercise of Project Control</b>	<b>108</b>
9.1	The Functions of Project Control and Oversight	109
9.2	The Requirements Phase	110
9.3	Contents of the Software Requirements Document	111
9.4	The Design Phase	113
9.5	The Implementation Phase	113
9.6	The Test and Integration Phase	115
9.7	Personnel Issues	116
9.8	The Hacker and Other Personalities	118
9.9	The Buck Stops at the Top	119
9.10	How People Think, Pay Attention, and Remember	121

<b>Chapter 10</b>	<b>The Development Process Methodology</b>	<b>125</b>
10.1	The “Design Hub” as Implementation Tool	126
10.2	The Architecture Definition Process	127
10.3	The Use of Large-Scale Representations	129
10.4	Design Team Meetings	130
10.5	Rapid Development versus Prototyping	131
10.6	The Traditional Development Methodology	132
10.7	Action Items, Change Requests, and Software Discrepancy Reports	134
10.8	Resolving Problems and Impasses	134
<b>Chapter 11</b>	<b>The Development of System Architectures</b>	<b>136</b>
11.1	Pushing the Architecture	137
11.2	The Point of “Acceptable Reality”	138
11.3	The Importance and Imperative of Visualizing Phenomena	140
11.4	Traditional Architectures	141
11.5	The Inferred Architecture	142
11.6	The Redesign or Upgrading of Existing Systems	144
11.7	The Approach to New Systems	145
<b>Chapter 12</b>	<b>The Impact of Leadership on Software Development</b>	<b>146</b>
12.1	Recognizing Good Leadership	146
12.2	The Concepts of Management and Leadership	148
12.3	Rewarding Failure	149
12.4	The Leader’s Subordinate	151
12.5	Indications of Poor Leadership	152
12.6	Leadership and Ethics	153
12.7	The Attributes of Leadership	153
12.7.1	Unselfishness	154
12.7.2	The Welfare of Others	154
12.7.3	Ambition	154
12.7.4	Integrity	154
12.7.5	Loyalty	155
12.7.6	Knowledge	155
12.7.7	Tact	156
12.7.8	Judgment	157
12.7.9	Initiative	157
12.7.10	Bearing	158
12.7.11	Courage	158
12.7.12	Decisiveness	159

12.7.13	Dependability	159
12.7.14	Dynamic Energy	160
12.7.15	Enthusiasm	161
12.7.16	Empowerment	163
12.8	The Ramifications of Failure	164
12.9	The Absence of Leadership	165
12.9.1	Absenteeism	166
12.9.2	Hidden Agendas	166
12.9.3	Communication Gap	167
12.9.4	Poorly Defined Goals	167
12.10	The Basis in Leadership for Failure	168
12.10.1	Personal Struggles	168
12.10.2	The “Machiavellian Prince”	169
12.11	The impact of Poor or Nonexistent Leadership	169
12.11.1	Conquering the Organization	170
<b>Chapter 13</b>	<b>Management of Software Systems Development</b>	<b>172</b>
13.1	Self-Respect in the Manager	173
13.2	The Ethical Workplace	173
13.3	Narcotics Use in the Workforce	174
13.4	Spotting Narcotics Addicts	177
13.5	Courage and Dynamic Energy in Management	178
13.6	The Traveling Manager	180
13.7	The Manager as Architect	181
13.8	The Phenomenon of Decision Making	182
13.9	The Concept of “Ability”	186
13.10	Manager: Administrator or Leader?	187
13.11	Authority, Responsibility, and Accountability	189
13.12	The Issue of Contempt	189
13.13	Management: The Fulcrum of Project Execution	191
13.14	The Ascendance of Mediocrity	191
13.15	The Pitfalls of Staffing Up	193
13.16	Salary Issues	195
13.17	Contracting Out Work	196
13.18	Evaluating Proposals	197
13.19	Cost Bidding too Early	198
<b>Chapter 14</b>	<b>Four Case Studies of Low-Cost Systems</b>	<b>200</b>
14.1	Case Study One: The Joint Theater Level Simulation (JTLS)	202
14.1.1	The Beginnings of JTLS	204
14.1.2	Estimating the Cost of War	205

- 14.1.3 Starting up the Effort 208
- 14.1.4 Costly Lessons Learned 209
- 14.2 Case Study Two: The Global Decision Support System (GDSS) 211
  - 14.2.1 GDSS System Size 211
  - 14.2.2 The History and Background of GDSS 212
  - 14.2.3 Expediting the System 213
  - 14.2.4 The Euler Sphere 214
  - 14.2.5 Beyond State of the Art 214
  - 14.2.6 A Replicated, Survivable, Synchronous Database Management System 214
  - 14.2.7 The Ultra Large Screen Display System 215
  - 14.2.8 The Local Area Networks 215
  - 14.2.9 The Wide Area Network 215
  - 14.2.10 Distributed Client/Server Technology 215
  - 14.2.11 Message Bus 215
  - 14.2.12 The GDSS Software Architecture 217
  - 14.2.13 Accepting the Challenge 220
  - 14.2.14 Initial Conditions 220
  - 14.2.15 Rapid Development: A Totally Different Approach 221
  - 14.2.16 There Can Be Only One! 223
  - 14.2.17 GDSS End-to-End Architecture 224
  - 14.2.18 Architecting the Development Effort 224
  - 14.2.19 Inferential Systems Architecture 225
  - 14.2.20 The GDSS System Software Layer 226
  - 14.2.21 Applications Language Selection 227
  - 14.2.22 Project Documentation 228
  - 14.2.23 Finding an Ada Expert 228
  - 14.2.24 Testing and Database Design 229
  - 14.2.25 Additional Difficulties 231
- 14.3 Case Study Three: The Topex TCCS System 233
  - 14.3.1 The Topex TCCS System 233
  - 14.3.2 System Description 233
  - 14.3.3 The Initial Conditions 235
  - 14.3.4 Project Constraints 235
  - 14.3.5 Implementation Considerations 236
  - 14.3.6 Development of TOPEX TCCS 237
  - 14.3.7 Agreeing to Do the Job 238
  - 14.3.8 Ground Truth 239
  - 14.3.9 Start of Project Development 239
  - 14.3.10 Architecting the Environment 241
  - 14.3.11 Hardware Procurement, Software Procurement 242
  - 14.3.12 The Relationship with the Contractor 244

14.3.13	Test Plan Scheduling	245	
14.3.14	Adherence to a Standard	246	
14.4	Case Study Four: The Jason 1 TCCS System (JTCCS)	246	
14.4.1	The Jason 1 TCCS System	247	
14.4.2	System Description	247	
14.4.3	The Initial Conditions	248	
14.4.4	Implementation Considerations	249	
14.4.5	The JTCCS Architecture	251	
<b>Chapter 15</b>	<b>Operations, Operators, and Users: Their Impact on Cost</b>		<b>257</b>
15.1	The Operational Requirement	258	
15.2	The Lack of an Operational Requirement	259	
15.3	The Operations Scenario	259	
15.4	The Cost of Operators and Analysts	260	
15.5	The Voyager Project Operations Center	261	
15.6	War Gaming	262	
15.7	The Value of Simulation	264	
15.8	Funds: A Perspective	264	
<b>Chapter 16</b>	<b>The Autonomous Cognitive System</b>		<b>266</b>
16.1	Introduction	266	
16.2	The Scale of Autonomy	267	
16.2.1	Category IV Autonomous Cognitive System: Superman	267	
16.2.2	Category III Autonomous Cognitive System: Perseus	268	
16.2.3	Category II Autonomous Cognitive System: Robot	269	
16.2.4	Category I Autonomous Cognitive System: Automaton	269	
16.3	“I Will, Because I Can”	270	
16.4	Toward Cognitive Dynamics	271	
16.5	Building an Autonomous System	271	
16.6	An Appropriate Model	272	
16.7	System-Level Requirements for Autonomy	273	
16.8	Architectural Domains for Autonomy	274	
16.8.1	Domain I: The Human Thought Architecture Model (Functional Architecture)	274	
16.8.2	Domain II: The Human Thought Process Model (Common Software Services)	275	
16.9	In Summary	277	

<b>Epilogue</b>	<b>279</b>
The Science of Computer Science	279
The Professional Software Manager	279
Cognitive Philosophy in a Modern Technical Context	280
Cognitive Dynamics is the Unifying Theory	281
The Issue of Software Cost	281
The Paradigm Shift of Cognitive Dynamics	282
<b>Glossary of Acronyms</b>	<b>283</b>
<b>Index</b>	<b>287</b>



# LIST OF FIGURES

- Figure 1** Computer Science Architectural Interpretation of Kantian Philosophy
- Figure 2** The Requirements and Design Process: Dialectically Achieving “Sufficient Reality”
- Figure 3** Project Organization: Traditional Hierarchical Approach: Isolated Control
- Figure 4** Project Organization: Flat, Nonhierarchical Approach: Optimal Control (Front View)
- Figure 5** Project Organization: Flat, Nonhierarchical Approach: Optimal Control (Overhead)
- Figure 6** Estimation of Software Cost: Software Work-Estimation Worksheet
- Figure 7** Two General Approaches to System Development: Known & Unknown Requirements
- Figure 8** The Kantian Thought Process: Decision-Making Schema
- Figure 9** Joint Theater Level Simulation: Operating System-dependent Architecture
- Figure 10** Global Decision Support System: Wide Area Network Configuration
- Figure 11** Global Decision Support System: Integrated Software Functional Design
- Figure 12** Global Decision Support System: Official Memo of Recognition
- Figure 13** Topex/Poseidon TCCS Software Architecture
- Figure 14** The Topex War Room: The Design Hub as an Implementation Environment
- Figure 15** JASON-1 TCCS Software Architecture
- Figure 16** JASON-1 TCCS Pluggable Architecture
- Figure 17** JASON-1 TCCS User Interface Client/Server Architecture

