

# Contents

<b>ACKNOWLEDGEMENTS.....</b>	<b>V</b>
<b>PREFACE.....</b>	<b>VII</b>
<b>PART I: THE CONTEXT OF MAINTENANCE.....</b>	<b>1</b>
OVERVIEW.....	1
DISCUSSION POINTS.....	2
<b>1. INTRODUCTION TO THE BASIC CONCEPTS.....</b>	<b>5</b>
1.1 INTRODUCTION.....	5
1.2 DEFINITIONS .....	6
1.3 THE BASICS.....	7
1.4 HOW NEW DEVELOPMENT AND MAINTENANCE ACTIVITIES DIFFER .....	9
1.5 WHY SOFTWARE MAINTENANCE IS NEEDED.....	10
1.6 MAINTAINING SYSTEMS EFFECTIVELY.....	11
1.7 CASE STUDY – AIR TRAFFIC CONTROL.....	12
1.8 CATEGORISING SOFTWARE CHANGE.....	14
1.9 SUMMARY .....	15
<b>2. THE MAINTENANCE FRAMEWORK.....</b>	<b>17</b>
2.1 INTRODUCTION.....	17
2.2 DEFINITIONS .....	17
2.3 A SOFTWARE MAINTENANCE FRAMEWORK.....	18
2.3.1 Components of the Framework.....	20
2.3.1.1 User .....	20
2.3.1.2 Environment.....	20
Operating environment.....	20
Organisational Environment.....	21
2.3.1.3 Maintenance Process .....	23
2.3.1.4 Software Product.....	25
2.3.1.5 Maintenance Personnel .....	28
2.3.2 Relations Between the Maintenance Factors .....	29
2.4 SUMMARY .....	31
<b>3. FUNDAMENTALS OF SOFTWARE CHANGE.....</b>	<b>33</b>
3.1 INTRODUCTION.....	33
3.2 DEFINITIONS .....	33
3.3 SOFTWARE CHANGE.....	34
3.3.1 Classification of Changes .....	34
3.3.1.1 Corrective Change.....	35

	3.3.1.2	Adaptive Change .....	36
	3.3.1.3	Perfective Change .....	36
	3.3.1.4	Preventive Change .....	39
	3.3.2	The Importance of Categorising Software Changes .....	40
	3.3.3	Case Study – The Need to Support an Obsolete System .....	40
	3.3.4	Incremental Release .....	41
3.4		ONGOING SUPPORT .....	42
3.5		LEHMAN’S LAWS .....	44
3.6		SUMMARY .....	46
<b>4.</b>		<b>LIMITATIONS AND ECONOMIC IMPLICATIONS TO SOFTWARE CHANGE .....</b>	<b>47</b>
4.1		INTRODUCTION .....	47
4.2		DEFINITIONS .....	47
4.3		ECONOMIC IMPLICATIONS OF MODIFYING SOFTWARE .....	48
4.4		LIMITATIONS TO SOFTWARE CHANGE .....	50
	4.4.1	Resource Limitations .....	50
	4.4.2	Quality of the Existing System .....	51
	4.4.3	Organisational Strategy .....	51
	4.4.4	Inertia .....	51
	4.4.5	Attracting and Retaining Skilled Staff .....	52
4.5		THE NOMENCLATURE AND IMAGE PROBLEMS .....	52
4.6		POTENTIAL SOLUTIONS TO MAINTENANCE PROBLEMS .....	54
	4.6.1	Budget and Effort Reallocation .....	54
	4.6.2	Complete Replacement of the System .....	55
	4.6.3	Maintenance of the Existing System .....	56
4.7		SUMMARY .....	56
<b>5.</b>		<b>THE MAINTENANCE PROCESS .....</b>	<b>59</b>
5.1		INTRODUCTION .....	59
5.2		DEFINITIONS .....	60
5.3		THE SOFTWARE PRODUCTION PROCESS .....	60
5.4		CRITICAL APPRAISAL OF TRADITIONAL PROCESS MODELS .....	65
	5.4.1	Code-and-Fix Model .....	66
	5.4.2	Waterfall Model .....	67
	5.4.3	Spiral Model .....	69
5.5		MAINTENANCE PROCESS MODELS .....	71
	5.5.1	Quick-Fix Model .....	76
	5.5.1.1	Case Study – Storage of Chronological Clinical Data .....	77
	5.5.2	Boehm’s Model .....	80
	5.5.3	Osborne’s Model .....	82
	5.5.4	Iterative Enhancement Model .....	84

5.5.5	Reuse-Oriented Model.....	85
5.6	WHEN TO MAKE A CHANGE.....	86
5.7	PROCESS MATURITY.....	87
5.7.1	Capability Maturity Model® for Software.....	88
5.7.2	Software Experience Bases.....	88
5.8	SUMMARY.....	89
<b>PART II: WHAT TAKES PLACE DURING MAINTENANCE.....</b>		<b>91</b>
	OVERVIEW.....	91
	DISCUSSION POINTS.....	94
<b>6.</b>	<b>PROGRAM UNDERSTANDING.....</b>	<b>97</b>
6.1	INTRODUCTION.....	98
6.2	DEFINITIONS.....	98
6.3	AIMS OF PROGRAM COMPREHENSION.....	100
6.3.1	Problem Domain.....	100
6.3.2	Execution Effect.....	101
6.3.3	Cause-Effect Relation.....	101
6.3.4	Product-Environment Relation.....	103
6.3.5	Decision-Support Features.....	103
6.4	MAINTAINERS AND THEIR INFORMATION NEEDS.....	103
6.4.1	Managers.....	104
6.4.2	Analysts.....	104
6.4.3	Designers.....	105
6.4.4	Programmers.....	105
6.5	COMPREHENSION PROCESS MODELS.....	107
6.6	MENTAL MODELS.....	109
6.7	PROGRAM COMPREHENSION STRATEGIES.....	110
6.7.1	Top-Down Model.....	111
6.7.2	Bottom-Up / Chunking Model.....	113
6.7.3	Opportunistic Model.....	115
6.8	READING TECHNIQUES.....	115
6.9	FACTORS THAT AFFECT UNDERSTANDING.....	116
6.9.1	Expertise.....	118
6.9.2	Implementation Issues.....	118
6.9.2.1	Naming Style.....	118
6.9.2.2	Comments.....	120
6.9.2.3	Decomposition Mechanism.....	121
6.9.3	Documentation.....	122
6.9.4	Organisation and Presentation of Programs.....	122
6.9.5	Comprehension Support Tools.....	125
6.9.5.1	Book Paradigm.....	125
6.9.6	Evolving Requirements.....	126

- 6.10 IMPLICATIONS OF COMPREHENSION THEORIES AND STUDIES ..... 128
  - 6.10.1 Knowledge Acquisition and Performance ..... 128
  - 6.10.2 Education and Training..... 129
  - 6.10.3 Design Principles ..... 129
  - 6.10.4 Guidelines and Recommendations..... 129
- 6.11 SUMMARY ..... 130
- 7. REVERSE ENGINEERING ..... 133**
  - 7.1 INTRODUCTION..... 133
  - 7.2 DEFINITIONS ..... 134
  - 7.3 ABSTRACTION ..... 134
    - 7.3.1 Function Abstraction ..... 135
    - 7.3.2 Data Abstraction ..... 135
    - 7.3.3 Process Abstraction ..... 135
  - 7.4 PURPOSE AND OBJECTIVES OF REVERSE ENGINEERING..... 135
  - 7.5 LEVELS OF REVERSE ENGINEERING ..... 138
    - 7.5.1 Redocumentation ..... 139
    - 7.5.2 Design Recovery..... 141
    - 7.5.3 Specification Recovery ..... 142
    - 7.5.4 Conditions for Reverse Engineering..... 143
  - 7.6 SUPPORTING TECHNIQUES..... 143
    - 7.6.1 Forward Engineering ..... 144
    - 7.6.2 Restructuring ..... 144
    - 7.6.3 Reengineering..... 146
  - 7.7 BENEFITS ..... 146
    - 7.7.1 Maintenance..... 146
    - 7.7.2 Software Reuse ..... 147
    - 7.7.3 Reverse Engineering and Associated Techniques in Practice ..... 147
  - 7.8 CASE STUDY: US DEPARTMENT OF DEFENSE INVENTORY..... 148
  - 7.9 CURRENT PROBLEMS ..... 149
  - 7.10 SUMMARY ..... 151
- 8. REUSE AND REUSABILITY ..... 153**
  - 8.1 INTRODUCTION..... 154
  - 8.2 DEFINITIONS ..... 154
  - 8.3 THE TARGETS FOR REUSE ..... 155
    - 8.3.1 Process ..... 155
    - 8.3.2 Personnel ..... 156
    - 8.3.3 Product..... 156
    - 8.3.4 Data..... 156
      - 8.3.4.1 Design ..... 156
      - 8.3.4.2 Program..... 157

8.4	OBJECTIVES AND BENEFITS OF REUSE .....	158
8.5	APPROACHES TO REUSE .....	159
8.5.1	Composition-Based Reuse.....	160
8.5.2	Generation-Based Reuse.....	162
8.5.2.1	Application Generator Systems.....	162
8.5.2.2	Transformation-Based Systems.....	163
8.5.2.3	Evaluation of the Generator-Based Systems .....	164
8.6	DOMAIN ANALYSIS .....	164
8.7	COMPONENTS ENGINEERING.....	166
8.7.1	Design for Reuse.....	166
8.7.1.1	Characteristics of Reusable Components .....	166
8.7.1.2	Problems with Reuse Libraries.....	168
8.7.2	Reverse Engineering.....	169
8.7.2.1	Case Study – Patient Identification .....	170
8.7.3	Components-Based Processes.....	171
8.8	REUSE PROCESS MODEL .....	172
8.8.1	Generic Reuse/Reusability Model .....	173
8.8.2	Accommodating a Reuse Process Model.....	176
8.9	FACTORS THAT IMPACT UPON REUSE .....	177
8.9.1	Technical Factors.....	177
8.9.1.1	Programming Languages.....	177
8.9.1.2	Representation of Information.....	177
8.9.1.3	Reuse Library .....	178
8.9.1.4	Reuse-Maintenance Vicious Cycle.....	178
8.9.2	Non-Technical Factors.....	178
8.9.2.1	Initial Capital Outlay.....	178
8.9.2.2	Not Invented Here Factor .....	179
8.9.2.3	Commercial Interest .....	179
8.9.2.4	Education .....	179
8.9.2.5	Project Co-ordination.....	179
8.9.2.6	Legal Issues.....	179
8.10	SUMMARY .....	181
<b>9.</b>	<b>TESTING .....</b>	<b>183</b>
9.1	INTRODUCTION.....	183
9.2	DEFINITIONS .....	183
9.3	WHY TEST SOFTWARE .....	184
9.4	WHAT IS A SOFTWARE TESTER’S JOB.....	186
9.5	WHAT TO TEST AND HOW .....	187
9.5.1	Who Chooses Test Data.....	187
9.6	CATEGORISING TESTS .....	189
9.6.1	Testing Code.....	190
9.6.1.1	Black Box and White Box Testing.....	190
9.6.1.2	Structured Testing .....	190
9.6.1.3	Integration Testing .....	191

- 9.6.1.4 Regression Testing ..... 191
- 9.7 VERIFICATION AND VALIDATION ..... 192
- 9.8 TEST PLANS ..... 192
  - 9.8.1 Points to Note ..... 193
- 9.9 CASE STUDY – THERAC 25..... 194
- 9.10 SUMMARY ..... 201
- 10. MANAGEMENT AND ORGANISATIONAL ISSUES ..... 203**
  - 10.1 INTRODUCTION..... 204
  - 10.2 DEFINITIONS ..... 205
  - 10.3 MANAGEMENT RESPONSIBILITIES ..... 205
  - 10.4 ENHANCING MAINTENANCE PRODUCTIVITY ..... 206
    - 10.4.1 Choosing the Right People ..... 206
    - 10.4.2 Motivating Maintenance Personnel ..... 206
    - 10.4.3 Communication..... 208
      - 10.4.3.1 Adequate Resources ..... 209
      - 10.4.3.2 Domain Knowledge..... 209
  - 10.5 MAINTENANCE TEAMS..... 210
    - 10.5.1 Temporary Team ..... 211
    - 10.5.2 Permanent Team ..... 211
  - 10.6 PERSONNEL EDUCATION AND TRAINING ..... 211
    - 10.6.1 Objectives ..... 212
      - 10.6.1.1 To Raise the Level of Awareness ..... 212
      - 10.6.1.2 To Enhance Recognition ..... 213
    - 10.6.2 Education and Training Strategies ..... 213
  - 10.7 ORGANISATIONAL MODES ..... 214
    - 10.7.1 Combined Development and Maintenance ..... 214
      - 10.7.1.1 Module Ownership..... 214
      - 10.7.1.2 Change Ownership ..... 215
      - 10.7.1.3 Work-Type..... 215
      - 10.7.1.4 Application-Type ..... 216
    - 10.7.2 Separate Maintenance Department ..... 216
  - 10.8 SUMMARY ..... 217
- PART III: KEEPING TRACK OF THE MAINTENANCE PROCESS ..... 219**
  - OVERVIEW ..... 219
  - DISCUSSION POINTS..... 220
- 11. CONFIGURATION MANAGEMENT..... 223**
  - 11.1 INTRODUCTION..... 223
  - 11.2 DEFINITIONS ..... 225
  - 11.3 CONFIGURATION MANAGEMENT..... 226
    - 11.3.1 A Specific View of Software Configuration Management ..... 231

11.3.1.1	Version Control.....	232
11.3.1.2	Building.....	234
11.3.1.3	Environment Management.....	234
11.3.1.4	Process Control.....	235
11.4	CHANGE CONTROL.....	235
11.4.1	The Responsibilities of Management in Change Control.....	236
11.5	DOCUMENTATION.....	238
11.5.1	Categories of Software Documentation.....	238
11.5.2	Role of Software Documentation.....	241
11.5.3	Producing and Maintaining Quality Documentation.....	242
11.6	SUMMARY.....	245
<b>12.</b>	<b>MAINTENANCE MEASURES.....</b>	<b>247</b>
12.1	INTRODUCTION.....	247
12.2	DEFINITIONS.....	248
12.3	THE IMPORTANCE OF INTEGRITY IN MEASUREMENT.....	249
12.3.1	Software Measurement.....	250
12.3.2	Software Measure and Software Metric.....	251
12.4	OBJECTIVES OF SOFTWARE MEASUREMENT.....	253
12.4.1	Evaluation.....	253
12.4.2	Control.....	253
12.4.3	Assessment.....	253
12.4.4	Improvement.....	254
12.4.5	Prediction.....	254
12.5	EXAMPLE MEASURES.....	254
12.5.1	Size.....	255
12.5.2	Complexity.....	255
12.5.2.1	McCabe's Cyclomatic Complexity.....	256
12.5.2.2	Halstead's Measures.....	257
12.5.3	Quality.....	259
12.5.3.1	Product Quality.....	259
12.5.3.2	Process Quality.....	259
12.5.4	Understandability.....	260
12.5.5	Maintainability.....	260
12.5.6	Cost Estimation.....	261
12.6	GUIDELINES FOR SELECTING MAINTENANCE MEASURES.....	261
12.7	SUMMARY.....	263
<b>PART IV:</b>	<b>BUILDING BETTER SYSTEMS.....</b>	<b>265</b>
OVERVIEW.....		265
DISCUSSION POINTS.....		266

<b>13. BUILDING AND SUSTAINING MAINTAINABILITY .....</b>	<b>269</b>
13.1 INTRODUCTION.....	270
13.2 DEFINITIONS .....	270
13.3 IMPACT ANALYSIS .....	271
13.3.1 Models and Strategies.....	271
13.3.2 Impact Analysis in Creating Maintainable Systems .....	272
13.4 QUALITY ASSURANCE.....	272
13.4.1 Fitness for Purpose .....	273
13.4.2 Correctness .....	274
13.4.3 Portability .....	274
13.4.4 Testability .....	275
13.4.5 Usability.....	275
13.4.5.1 Case Study – Usability .....	275
13.4.6 Reliability .....	276
13.4.7 Efficiency.....	277
13.4.8 Integrity .....	277
13.4.9 Reusability.....	278
13.4.10 Interoperability .....	278
13.5 FOURTH-GENERATION LANGUAGES.....	279
13.5.1 Properties of Fourth-Generation Languages .....	281
13.5.2 Impact on Maintenance.....	282
13.5.2.1 Increased Productivity.....	282
13.5.2.2 Reduction in Cost.....	283
13.5.2.3 Ease of Understanding .....	283
13.5.2.4 Automatic Documentation .....	283
13.5.2.5 Reduction in Workload .....	283
13.5.3 Weaknesses of Fourth-Generation Languages.....	283
13.5.3.1 Application-Specific.....	284
13.5.3.2 Proprietary.....	284
13.5.3.3 Hyped Ease of Use .....	284
13.5.3.4 Poor Design.....	284
13.6 OBJECT-ORIENTED PARADIGMS .....	285
13.6.1 Decomposition to Aid Comprehension.....	286
13.6.2 Impact on Maintenance.....	288
13.6.3 Migration to Object-Oriented Platforms .....	290
13.6.4 Approaches .....	290
13.6.5 Retraining Personnel.....	291
13.7 OBJECT-ORIENTED TECHNIQUES IN SOFTWARE MAINTENANCE ....	292
13.7.1 Case Study – Mobile2000 .....	292
13.7.2 Case Study – Insight II .....	293
13.7.3 Case Study – Image Filing System .....	295
13.8 SUMMARY .....	297

**14. MAINTENANCE TOOLS.....299**

14.1 INTRODUCTION..... 299

14.2 DEFINITIONS ..... 300

14.3 CRITERIA FOR SELECTING TOOLS..... 300

14.4 TAXONOMY OF TOOLS ..... 302

14.5 TOOLS FOR COMPREHENSION AND REVERSE ENGINEERING ..... 302

    14.5.1 Program Slicer ..... 303

    14.5.2 Static Analyser..... 303

    14.5.3 Dynamic Analyser ..... 304

    14.5.4 Data Flow Analyser ..... 304

    14.5.5 Cross-Referencer ..... 304

    14.5.6 Dependency Analyser ..... 305

    14.5.7 Transformation Tool..... 305

14.6 TOOLS TO SUPPORT TESTING ..... 305

    14.6.1 Simulator ..... 305

    14.6.2 Test Case Generator..... 306

    14.6.3 Test Paths Generator..... 306

14.7 TOOLS TO SUPPORT CONFIGURATION MANAGEMENT ..... 306

    14.7.1 Source Code Control System..... 307

    14.7.2 Other Utilities ..... 308

14.8 OTHER TASKS ..... 308

    14.8.1 Documentation..... 308

    14.8.2 Complexity Assessment..... 308

14.9 SUMMARY ..... 309

**PART V: LOOKING TO THE FUTURE..... 311**

OVERVIEW..... 311

THE PAST AND PRESENT..... 312

RESEARCH AREAS ..... 313

    Classification..... 313

    Software Experience Bases ..... 313

    Software Reuse ..... 313

    Support Tools..... 314

    Software Measurement ..... 314

    Program Comprehension..... 314

    The Software Maintenance Process ..... 315

    The Threesome Marriage ..... 315

THE BEST OF BOTH WORLDS..... 316

**REFERENCES ..... 317**

**INDEX..... 341**