
Contents

ACKNOWLEDGMENTS	xi
ACRONYMS	xiii
INTRODUCTION.	xvii
CHAPTER 1. SOFTWARE MODERNIZATION:	
A BUSINESS VISION	1
1.1. Software-based business.	1
1.2. Information-driven business	2
1.2.1. Adaptation to business	4
1.3. The case of tourism industry	7
1.4. IT progress acceleration	11
1.5. Legacy world	13
1.5.1. Exiting the legacy world	15
1.5.2. Legacy world professionals	16
1.6. Conclusions	18
CHAPTER 2. SOFTWARE MODERNIZATION:	
TECHNICAL ENVIRONMENT	21
2.1. Legacy system.	21
2.2. Modernization.	22
2.2.1. Replacement	24
2.2.2. Migration	25
2.2.3. Modernization versus migration.	27

2.2.4. The superiority of white-box modernization	29
2.3. Software engineering principles underpinning modernization	31
2.3.1. Re-engineering in action	33
2.3.2. Re-engineering challenges	36
2.4. Conclusions	37
CHAPTER 3. STATUS OF COBOL LEGACY APPLICATIONS	39
3.1. OLTP versus batch programs	41
3.2. Mainframes	42
3.3. Data-driven design	43
3.4. COBOL degeneration principle	44
3.5. COBOL pitfalls	46
3.6. Middleware for COBOL	47
3.7. Moving COBOL OLTP/batch programs to Java	49
3.8. COBOL is not a friend of Java, and vice versa.	51
3.9. Spaghetti code.	52
3.9.1. Spaghetti code sample.	53
3.9.2. Code comprehension	56
3.10. No longer COBOL?	57
3.11. Conclusions	58
CHAPTER 4. SERVICE-ORIENTED ARCHITECTURE (SOA)	59
4.1. Software architecture <i>versus</i> information system urbanization.	59
4.2. Software architecture evolution	60
4.3. COBOL own style of software architecture	61
4.4. The one-way road to SOA.	64
4.5. Characterization of SOA.	66
4.5.1. Preliminary note	66
4.5.2. From objects to components and services	66
4.5.3. Type versus instance	67
4.5.4. Distribution concerns	68
4.5.5. Functional grouping	68
4.5.6. Granularity	69

4.5.7. Technology-centrism	70
4.5.8. Composition at design time (... is definitely modeling)	72
4.5.9. Composition at runtime	77
4.6. Conclusions	78
CHAPTER 5. SOA IN ACTION	79
5.1. Service as materialized component	81
5.2. Service as Internet resource	85
5.2.1. Pay-per-use service	87
5.2.2. Free service	89
5.2.3. Data feed service	90
5.3. High-end SOA	93
5.4. SOA challenges	95
5.5. The Cloud	97
5.5.1. COBOL in the Cloud	98
5.5.2. Computing is just resource consumption	99
5.5.3. Cloud computing is also resource consumption, but...	101
5.5.4. Everything as a service	102
5.5.5. SOA in the Cloud	104
5.5.6. The cloud counterparts	105
5.6. Conclusions	106
CHAPTER 6. MODEL-DRIVEN DEVELOPMENT (MDD)	109
6.1. Why MDD?	110
6.2. Models, intuitively	111
6.3. Models, formally	112
6.4. Models as computerized objects	113
6.5. Model-based productivity	118
6.6. Openness through standards	118
6.6.1. Model-Driven Architecture (MDA)	120
6.7. Models and people	121
6.8. Metamodeling	123
6.8.1. Metamodeling, put simply	123
6.9. Model transformation	125
6.10. Model transformation by example	125

6.11. From contemplative to executable models	126
6.12. Model execution in action	127
6.13. Toward Domain-Specific Modeling Languages (DSMLs)	129
6.14. Conclusions	132
CHAPTER 7. MODEL-DRIVEN SOFTWARE MODERNIZATION	135
7.1. Reverse and forward engineering are indivisible components of modernization	137
7.2. Architecture-Driven Modernization (ADM)	138
7.3. ASTM and KDM at a glance	142
7.4. Variations on ASTM	146
7.5. From ASTM to KDM	148
7.6. Variations on KDM	149
7.7. Automation	153
7.8. Conclusions	153
CHAPTER 8. SOFTWARE MODERNIZATION METHOD AND TOOL	155
8.1. BLU AGE overview	156
8.2. The toolbox	158
8.2.1. BLU AGE format required for forward engineering	160
8.2.2. Reverse tooling	162
8.3. BLU AGE as an ADM- and MDA-compliant tool	170
8.4. Modernization workflow	173
8.4.1. Initialization	173
8.4.2. Realization	182
8.4.3. Validation and deployment	187
8.5. Conclusions	188
CHAPTER 9. CASE STUDY	191
9.1. Case study presentation	192
9.2. Legacy modernization in action	195
9.2.1. Creating modernization project	196
9.2.2. Better dealing with the legacy material	196
9.2.3. Strategy for modernizing screens	202

9.2.4. Strategy for modernizing data items	203
9.2.5. Creating forward project	204
9.2.6. Entity extraction	207
9.2.7. From screens to pages and UI components	209
9.3. Annotations	209
9.4. Pattern definition	211
9.4.1. Pattern for simple statements	211
9.4.2. Patterns for operation calls	213
9.4.3. Patterns for operation calls with arguments	214
9.4. Database exchange modernization	216
9.5. Transmodeling	219
9.6. Transmodeling complex functionalities	226
9.6.1. Transmodeling the “custCost” program	228
9.6.2. Modernizing “Add a new reservation”	233
9.7. Application generation and testing	234
9.8. Conclusions	235
BIBLIOGRAPHY	239
INDEX	243