
Contents

From: *Handbook of Requirements and Business Analysis*, by
Bertrand Meyer, Springer, 2022. © Bertrand Meyer, 2022.
See book page at requirements.bertrandmeyer.com.

PREFACE	VII
The material	vii
Obstacles to quality	viii
Descriptive and prescriptive	viii
A balanced view	x
Key ideas	xi
Geek and non-geek	xiii
The author's experiences behind this handbook	xiii
Bibliographical notes and further reading	xv
Acknowledgments	xv
Credits	xviii
CONTENTS	XIX
1 REQUIREMENTS: BASIC CONCEPTS AND DEFINITIONS	1
1.1 Dimensions of requirements engineering	1
1.1.1 Universe of discourse: the four PEGS	1
1.1.2 Distinguishing system and environment	2
1.1.3 The organizations involved	2
1.1.4 Stakeholders	3
1.2 Defining requirements	4
1.2.1 Properties	4
1.2.2 Statements	4
1.2.3 Relevance	5
1.2.4 Requirement	5
1.2.5 Requirements engineering, business analysis	6
1.3 Kinds of requirements element	6
1.4 Requirements affecting goals	6
1.4.1 Goal	6
1.4.2 Special case: obstacle	6
1.5 Requirements on the project	8
1.5.1 Task	8
1.5.2 Product	8
1.6 Requirements on the system	8
1.6.1 Behavior	8
1.6.2 Special cases: functional and non-functional requirements	8
1.6.3 Special cases: examples (scenarios)	8

1.7 Requirements on the environment	9
1.7.1 Constraint	9
1.7.2 Special cases of constraints: business rule, physical rule, engineering decision	9
1.7.3 Assumption	9
1.7.4 Distinguishing between constraints and assumptions	10
1.7.5 Effect	10
1.7.6 Invariant	10
1.8 Requirements applying to all dimensions	11
1.8.1 Component	11
1.8.2 Responsibility	11
1.8.3 Limit	11
1.8.4 Special case: role	11
1.9 Special requirements elements	12
1.9.1 Silence	12
1.9.2 Noise	12
1.9.3 Special case: hint	12
1.9.4 Metarequirement	13
1.9.5 Special case: justification	13
1.10 The people behind requirements	13
1.10.1 Categories of stakeholders	13
1.10.2 Who produces requirements?	16
1.11 Why perform requirements?	17
1-E Exercises	18
Bibliographical notes and further reading	19
Terminology note: verification and validation	20
2 REQUIREMENTS: GENERAL PRINCIPLES	21
2.1 What role for requirements?	21
2.1.1 The need for requirements	21
2.1.2 The role of requirements	22
2.1.3 The nature of requirements	22
2.1.4 The evolution of requirements	23
2.1.5 The place of requirements in the project lifecycle	25
2.1.6 The form of requirements	26
2.1.7 Outcomes of requirements	27
2.2 Human aspects	28
2.2.1 Stakeholders	28
2.2.2 Authors	28
2.3 Requirements elicitation and production	29
2.4 Requirements management	30
2.5 Requirements quality	31
2.6 Other principles	32
2-E Exercises	33
Bibliographical notes and further reading	33

3 STANDARD PLAN FOR REQUIREMENTS	35
3.1 Overall structure	35
3.2 Front and back matter	36
3.3 Using the plan	36
3.3.1 Forms of requirements conforming to the Standard Plan	36
3.3.2 Customizing the plan	37
3.3.3 Mutual references	37
3.4 The Goals book	38
3.5 The Environment book	40
3.6 The System book	42
3.7 The Project book	43
3.8 Minimum requirements	45
3-E Exercises	45
Bibliographical notes and further reading	46
4 REQUIREMENTS QUALITY AND VERIFICATION	47
4.1 Correct	48
4.1.1 About correctness	48
4.1.2 Ensuring correctness	48
4.1.3 Assessing correctness	48
4.1.4 Parts of the Standard Plan particularly relevant to assessing correctness	49
4.2 Justified	49
4.2.1 About justifiability	49
4.2.2 Ensuring justifiability	50
4.2.3 Assessing justifiability	50
4.2.4 Parts of the Standard Plan particularly relevant to assessing justifiability	51
4.3 Complete	51
4.4 Consistent	52
4.4.1 About consistency	52
4.4.2 Ensuring consistency	52
4.4.3 Assessing consistency	53
4.4.4 Parts of the Standard Plan particularly relevant to assessing consistency	53
4.5 Unambiguous	54
4.5.1 About non-ambiguity	54
4.5.2 Ensuring non-ambiguity	54
4.5.3 Assessing non-ambiguity	54
4.5.4 Parts of the Standard Plan particularly relevant to assessing non-ambiguity	54
4.6 Feasible	55
4.6.1 About feasibility	55
4.6.2 Ensuring feasibility	55
4.6.3 Assessing feasibility	56
4.6.4 Parts of the Standard Plan particularly relevant to assessing feasibility	56

4.7 Abstract	56
4.7.1 About abstractness	56
4.7.2 The difficulty of abstracting	57
4.7.3 Overspecification	59
4.7.4 Design and implementation hints	59
4.7.5 Beware of use cases	60
4.7.6 Ensuring abstractness	60
4.7.7 Assessing abstractness	60
4.7.8 Parts of the Standard Plan particularly relevant to assessing abstractness	60
4.8 Traceable	61
4.8.1 About traceability	61
4.8.2 Ensuring traceability	61
4.8.3 Assessing traceability	62
4.8.4 Parts of the Standard Plan particularly relevant to assessing traceability	62
4.9 Delimited	62
4.9.1 About delimitation	62
4.9.2 Ensuring delimitation	63
4.9.3 Assessing delimitation	63
4.9.4 Parts of the Standard Plan particularly relevant to assessing delimitation	63
4.10 Readable	63
4.10.1 About readability	63
4.10.2 Ensuring readability	64
4.10.3 Assessing readability	64
4.10.4 Parts of the Standard Plan particularly relevant to assessing readability	65
4.11 Modifiable	65
4.11.1 About modifiability	65
4.11.2 Ensuring modifiability	65
4.11.3 Assessing modifiability	65
4.11.4 Parts of the Standard Plan particularly relevant to assessing modifiability	65
4.12 Verifiable	66
4.12.1 About verifiability	66
4.12.2 Ensuring verifiability	66
4.12.3 Assessing (“verifying”) verifiability	66
4.12.4 Parts of the Standard Plan particularly relevant to assessing verifiability	66
4.13 Prioritized	67
4.13.1 About prioritization	67
4.13.2 Ensuring prioritization	67
4.13.3 Assessing prioritization	67
4.13.4 Parts of the Standard Plan particularly relevant to assessing prioritization	67
4.14 Endorsed	68
4.14.1 About endorsement	68
4.14.2 Ensuring endorsement	68
4.14.3 Assessing endorsement	68
4.14.4 Parts of the Standard Plan particularly relevant to assessing endorsement	68
4-E Exercises	69
Bibliographical notes and further reading	70

5 HOW TO WRITE REQUIREMENTS	71
5.1 When and where to write requirements	71
5.2 The seven sins of the specifier	72
5.2.1 The Sins list	72
5.2.2 Noise and silence	73
5.2.3 Remorse	73
5.2.4 Falsehood	74
5.2.5 Synonyms	74
5.2.6 Etcetera lists	74
5.3 Repetition	75
5.4 Binding and explanatory text	77
5.5 Notations for requirements	79
5.5.1 Natural language	79
5.5.2 Graphical notations	80
5.5.3 Formal notations	82
5.5.4 Tabular notations	83
5.5.5 Combining notations	84
5.6 Some examples: bad, less bad, good	85
5.6.1 “Provide status messages”	85
5.6.2 The flashing editor	86
5.6.3 Always an error report?	86
5.6.4 Words to avoid	87
5.7 Style rules for natural-language requirements	88
5.7.1 General guidelines	88
5.7.2 Use correct spelling and grammar	89
5.7.3 Use simple language	90
5.7.4 Identify every part	90
5.7.5 Be consistent	91
5.7.6 Be prescriptive	91
5.8 The TBD rule	92
5.9 Documenting goals	93
5.10 The seven sins: a classic example	93
5.10.1 A simple specification	94
5.10.2 A detailed description	95
5.10.3 More ambiguity!	100
5.10.4 Lessons from the example	101
5.10.5 OK, but can we do better?	102
5-E Exercises	102
Bibliographical notes and further reading	103

6 HOW TO GATHER REQUIREMENTS	105
6.1 Planning and documenting the process	105
6.2 The role of stakeholders	105
6.3 Sources other than stakeholders	106
6.4 The glossary	107
6.4.1 Clarify the terminology	108
6.4.2 Kidnapped words	108
6.4.3 Acronyms	109
6.5 Assessing stakeholders	109
6.6 Making business analysts and domain experts work together	111
6.7 Biases, interviews and workshops	112
6.8 Conducting effective interviews	113
6.8.1 Setting up and conducting an interview	113
6.8.2 Interview reports	114
6.9 Conducting effective workshops	114
6.9.1 Why workshops help	114
6.9.2 When to run workshops	115
6.9.3 Planning a workshop	115
6.9.4 Running a workshop	116
6.9.5 After the workshop	117
6.10 Asking the right questions	118
6.10.1 Uncover the unsaid	118
6.10.2 Cover all PEGS	118
6.10.3 Do not confuse roles	119
6.10.4 Ask effective questions	119
6.10.5 Get stakeholders to prioritize	121
6.11 Prototypes: tell or show?	122
6.11.1 What is a prototype?	122
6.11.2 Incremental prototypes	122
6.11.3 Throwaway prototypes	123
6.11.4 UI prototypes	123
6.11.5 Feasibility prototypes	123
6.11.6 Limitations of prototypes	125
6.11.7 Risk assessment and mitigation	126
6-E Exercises	126
Bibliographical notes and further reading	127

7 SCENARIOS: USE CASES, USER STORIES	129
7.1 Use cases	129
7.2 User stories	132
7.3 Epics and use case slices	133
7.4 The benefits of scenarios for requirements	133
7.5 The limitations of scenarios for requirements	134
7.6 The role of use cases and user stories in requirements	135
7-E Exercises	136
Bibliographical notes and further reading	136
8 OBJECT-ORIENTED REQUIREMENTS	137
8.1 Two kinds of system architecture	137
8.2 The notion of class	138
8.3 Relations between classes and the notion of deferred class	139
8.4 Why object-oriented requirements?	140
8.5 An OO notation	142
8.6 Avoiding premature ordering	143
8.6.1 The limitations of sequential ordering	143
8.6.2 A detour through stacks	144
8.7 Logical constraints versus premature ordering	147
8.7.1 A contract-based specification	147
8.7.2 Logical constraints are more general than sequential orderings	150
8.7.3 What use for scenarios?	151
8.7.4 Where do scenarios fit?	151
8.7.5 Different roles for different techniques	152
8.7.6 Towards formal methods and abstract data types	153
8.7.7 “But it’s design!”	153
8.7.8 Towards seamlessness	154
8.8 The seven kinds of class	154
8.8.1 Requirements classes	155
8.8.2 Design and implementation classes	156
8.8.3 Goals and project classes	156
8.8.4 Permissible relations between classes	156
8.9 Going object-oriented	158
8-E Exercises	159
Bibliographical notes and further reading	159

9	BENEFITING FROM FORMAL METHODS	161
9.1	Those restless Swiss!	161
9.2	Basic math for requirements	162
9.2.1	Logic and sets	162
9.2.2	Operations on sets	163
9.2.3	Relations	163
9.2.4	Functions	164
9.2.5	Powers and closures	165
9.2.6	Sequences	166
9.3	The relocating population, clarified	167
9.3.1	Naming components of a specification	167
9.3.2	Interpretation 1	167
9.3.3	Interpretation 2	168
9.3.4	Back to English: the formal picnic	168
9.4	Who writes formal specifications?	170
9.5	An example: text formatting, revisited	171
9.5.1	Defining a framework	171
9.5.2	The distinctive nature of requirements	173
9.5.3	Text formatting as minimization	174
9.5.4	The specification	175
9.5.5	Analyzing the specification	175
9.5.6	Proving requirements properties	176
9.5.7	Back from the picnic	178
9.5.8	Error handling	179
9.6	Formal requirements languages	180
9.7	Expressing formal requirements in a programming language	182
9-E	Exercises	183
	Bibliographical notes and further reading	185
10	ABSTRACT DATA TYPES	187
10.1	An example	187
10.2	The concept of abstract data type	188
10.3	Functions and their signatures	188
10.4	Axioms	190
10.5	ADT expressions as a model of computation	190
10.6	Sufficient completeness	191
10.6.1	A workable notion of completeness	192
10.6.2	A proof of sufficient completeness	193
10.7	Partial functions and preconditions	194
10.7.1	The need for partial functions	194
10.7.2	Partial functions in ADT specifications	194
10.7.3	The nature of preconditions	195
10.7.4	Expression correctness	196

10.7.5 Ascertaining correctness	197
10.7.6 No vicious cycle	197
10.8 Using abstract data types for requirements	198
10.8.1 Turning an ADT into a class	198
10.8.2 Functional and imperative styles	199
10.8.3 From an ADT to a class	200
10.9 ADTs: lessons for the requirements practitioners	200
10-E Exercises	201
Bibliographical notes and further reading	203
11 ARE MY REQUIREMENTS COMPLETE?	205
11.1 Document completeness	205
11.2 Goal completeness	206
11.3 Scenario completeness	207
11.4 Environment completeness	208
11.5 Interface completeness	208
11.6 Command-query completeness	209
Bibliographical notes and further reading	210
12 REQUIREMENTS IN THE SOFTWARE LIFECYCLE	211
12.1 Rescuing the Waterfall	211
12.2 Rescuing the Spiral model	212
12.3 Rescuing RUP	214
12.4 Rescuing Agile and DevOps	215
12.4.1 An agile lifecycle	215
12.4.2 Agile damage, agile benefit	216
12.4.3 DevOps	216
12.5 The Cluster model	217
12.6 Seamless development	218
12.6.1 The unity of software development	218
12.6.2 A seamless process	219
12.6.3 Reversibility	220
12.7 A unifying model	221
12.7.1 Overall iterative scheme	221
12.7.2 Not all sprints are created equal	221
12.7.3 An example sequence of sprints	223
12.7.4 Implement early and often	224
12.7.5 Detailed view of a sprint	225
12.7.6 A combination of best practices	226
Bibliographical notes and further reading	227
BIBLIOGRAPHY	229
INDEX	239