

Table of Contents

Preface	xi
Chapter 1. Why Use UML to Model Network Protocols?	1
1.1. Modeling network protocols	1
1.1.1. The complexity of communication protocols	1
1.1.2. Traditional modeling	3
1.1.2.1. Basic models and formal description techniques	4
1.1.2.2. Automata and state machine models	4
1.1.2.3. Petri Net models	10
1.1.2.4. Formal and semi-formal languages	11
1.1.2.5. Towards a new modeling language	12
1.1.3. Traditional validation	12
1.1.4. Need for a unified language for description, validation and simulation	13
1.2. UML as a common language	14
1.2.1. Overview	14
1.2.2. The beginning	15
1.2.3. Brief review	16
1.2.3.1. Class diagram	16
1.2.3.2. Package diagram	18
1.2.3.3. Object diagram	19
1.2.3.4. Component diagram	19
1.2.3.5. Composite structure diagram	20
1.2.3.6. Deployment diagram	20
1.2.3.7. Sequence diagram	21
1.2.3.8. Profile diagram	22

1.2.3.9. Activity diagram	22
1.2.3.10. Communication diagram	23
1.2.3.11. Interaction overview diagram	23
1.2.3.12. Timing diagram.	24
1.2.3.13. Use case diagram.	25
1.2.3.14. State machine diagram	26
1.2.4. UML for network protocols.	26
1.2.5. Some general UML tools	27
1.3. Chapter summary	28
1.4. Bibliography	28
Chapter 2. Simple Transmission	31
2.1. Introduction.	31
2.2. Echo	31
2.2.1. Requirement specification.	31
2.2.2. Analysis	32
2.2.2.1. Sequence diagram	32
2.2.2.2. Concerned classes	32
2.2.2.3. Signals list definition	33
2.2.3. Architecture design.	36
2.2.4. Detailed design: class behavior.	37
2.2.4.1. Client class	38
2.2.4.2. Server class	39
2.2.5. Echo: validation and simulation	40
2.2.5.1. Simulating the model	41
2.2.5.2. Model validation.	42
2.3. Unidirectional: simple data sending	42
2.3.1. Requirement specification.	43
2.3.2. Analysis	45
2.3.2.1. Sequence diagram	45
2.3.2.2. Signals list definition	46
2.3.3. Architecture design.	46
2.3.4. Detailed design	47
2.3.4.1. First variation: protocol notation (transition oriented).	47
2.3.4.2. Second variation: behavior notation (state oriented).	48
2.3.5. Validation and simulation	50
2.3.5.1. Simulating the model	50
2.3.5.2. Model validation.	51
2.4. Full duplex: simple data sending	52

2.4.1. Specification	52
2.4.2. Analysis	53
2.4.2.1. Sequence diagram	53
2.4.2.2. Concerned classes	54
2.4.3. Architecture design	56
2.4.4. Validation and simulation	59
2.4.4.1. Simulating the model	59
2.4.4.2. Model validation	62
2.4.5. Different ways of doing the same thing	62
2.4.5.1. A single machine	62
2.4.5.2. Interruption	66
2.4.5.3. Composite states with concurrent regions	70
2.4.5.4. Parallel threads	72
2.5. Chapter summary	73
2.6. Bibliography	73
Chapter 3. Simple Chat Application	75
3.1. Introduction	75
3.2. Requirements	75
3.3. Analysis	76
3.3.1. Sequence diagrams	78
3.3.1.1. Splitting the system up	80
3.3.1.2. Open session	81
3.3.1.3. Timeout	81
3.3.1.4. Refuse	82
3.3.1.5. Accept	83
3.3.1.6. Close	83
3.3.1.7. Send data	84
3.3.2. Concerned classes	85
3.3.3. Signal list definition	86
3.4. Architecture design	88
3.5. Detailed design	89
3.5.1. Open session	90
3.5.2. Timeout	92
3.5.3. Refusing the session	93
3.5.4. Accepting session	94
3.5.5. Closing session	95
3.5.6. Sending data	96
3.5.6.1. Case 1: Successful delivery	96

3.5.6.2. Case 2: the receiving user closes the session	98
3.5.6.3. Case 3: The whole session is closed	100
3.6. Simple chat simulation	101
3.6.1. Intuitive test	101
3.6.2. Initial tests	102
3.6.2.1. Refuse session	103
3.6.2.2. Accept session	104
3.6.2.3. Close session	105
3.6.2.4. Send data	106
3.6.2.5. Coverage statistics	108
3.6.3. Extended tests (fault tolerance)	110
3.6.3.1. Proposed scenarios	111
3.6.3.2. Send when session is closed	111
3.6.3.3. A-Sends while B-Closes	112
3.6.3.4. Close (with session closed)	115
3.6.3.5. Open (when session is already opened)	119
3.6.3.6. A-Opens – B-Opens	123
3.6.3.7. A-Closes – B-Closes	126
3.6.3.8. A-sends – B-sends	128
3.7. Chapter summary	130
3.8. Bibliography	131
Chapter 4. Non-reliable Transmission Mediums	133
4.1. Introduction	133
4.2. Requirements	134
4.3. Analysis	135
4.3.1. Sequence diagrams	138
4.3.1.1. Configure	138
4.3.1.2. Send	139
4.3.2. Concerned classes	141
4.3.2.1. Accessory classes	142
4.3.2.2. Class diagram	143
4.3.3. Signal list definition	144
4.4. Architecture design	147
4.4.1. Detailing the sequence diagram	148
4.5. Detailed design	150
4.5.1. Random class	151
4.5.1.1. ini operation	152
4.5.1.2. rand operation	152

4.5.1.3. randAB operation	153
4.5.2. Losing class	153
4.5.2.1. setLossesRate operation	154
4.5.2.2. Initialize operation.	154
4.5.3. Delaying class	155
4.5.3.1. setMaxDelay operation	156
4.5.3.2. Initialize operation.	156
4.5.4. Non-reliable medium class	158
4.5.4.1. Initialize operation.	158
4.6. Validation.	160
4.6.1. Configure.	160
4.6.2. Send.	162
4.6.2.1. Neither losses nor delay.	162
4.6.2.2. Losing everything	164
4.6.2.3. No losses and high delay	165
4.6.2.4. Losses and delay	167
4.6.3. Extending the model	170
4.6.3.1. Adding a payload	170
4.6.3.2. Consecutive data packets	174
4.7. Chapter summary	179
4.8. Bibliography	179
Chapter 5. Simple Transport Protocol	181
5.1. Introduction.	181
5.2. Requirements.	182
5.3. The Alternating Bit Protocol.	182
5.3.1. Basic communication features with Petri Nets.	182
5.3.2. The algorithm	188
5.4. Analysis	191
5.4.1. First step: The Alternating Bit Protocol.	191
5.4.2. Second step: Linking the application	194
5.4.3. Third step: Linking the medium	196
5.4.4. Concerned classes	198
5.4.5. Signal list definition	199
5.5. Architecture design	200
5.5.1. Simple transport protocol	200
5.5.2. Simulation model	203
5.6. Detailed design	204
5.6.1. ADU data type	204

5.6.2. PDU data type	205
5.6.3. ADU_Manager	206
5.6.4. PDU_Manager	208
5.6.5. ABP_Sender	210
5.6.6. ABP_Receiver	215
5.7. Simulations	217
5.7.1. Initialization and configuration.	217
5.7.2. No losses	219
5.7.2.1. Open session request without answer.	219
5.7.2.2. Open session request with answer.	224
5.7.2.3. Send data	226
5.7.3. Losses.	229
5.8. Further considerations	235
5.8.1. Acknowledgements at different levels	235
5.8.2. Delayed acknowledgements.	237
5.9. Chapter summary	238
5.10. Bibliography	239
Appendix. Detailed Diagrams of the Simple Transport Protocol.	241
A.1. State machines for the Application Data Unit Manager (Simple Transport Protocol).	242
A.2. Detailed simulations of the Simple Transport Protocol.	245
A.2.1. Initializing and configuring the medium	245
A.2.2. No losses	246
A.2.2.1. Open session request without answer	246
A.2.2.2. Open session request with answer	248
A.2.2.3. Send data	249
A.2.3. Losses	250
A.2.4. Losses and delay	253
Index	259