THE JR PROGRAMMING LANGUAGE: CONCURRENT PROGRAMMING IN AN EXTENDED JAVA

RONALD A. OLSSON
Department of Computer Science
University of California, Davis

AARON W. KEEN
Department of Computer Science
California Polytechnic State University

Kluwer Academic Publishers
Boston/Dordrecht/London

Contents

Dedication v
List of Figures xv
List of Tables xvii
Preface xix
Acknowledgments xxv

1. INTRODUCTION 1
   - 1.1 Key JR Components 3
   - 1.2 Two Simple Examples 4
   - 1.3 Matrix Multiplication 6
   - 1.4 Concurrent File Search 8
   - 1.5 Critical Section Simulation 10
   - 1.6 Translating and Executing JR Programs 12
   - 1.7 Vocabulary and Notation 13
   - Exercises 13

Part I Extensions for Concurrency

2. OVERVIEW OF EXTENSIONS 17
   - 2.1 Process Interactions via Operations 17
   - 2.2 Distributing JR Programs 19

3. OP-METHODS, OPERATIONS, AND CAPABILITIES 21
   - 3.1 Op-methods 21
   - 3.2 Operation and Method Declarations 22
   - 3.3 Operation Capabilities 22

Exercises 25

4. CONCURRENT EXECUTION 27
4.1 Process Declarations 27
4.2 The Unabbreviated Form of Processes 31
4.3 Static and Non-static Processes 34
4.4 Process Scheduling and Priorities 35
4.5 Automatic Termination Detection 36
Exercises 38

5. SYNCHRONIZATION USING SHARED VARIABLES 43
5.1 The Critical Section Problem 43
5.2 An Incorrect Solution 45
5.3 An Alternating Solution 46
5.4 The Bakery Algorithm for Two Processes 47
5.5 The Bakery Algorithm for N Processes 49
Exercises 50

6. SEMAPHORES 53
6.1 Semaphore Declarations and Operations 53
6.2 The Dining Philosophers Problem 56
6.3 Barrier Synchronization 58
Exercises 61

7. ASYNCHRONOUS MESSAGE PASSING 65
7.1 Operations as Message Queues 65
7.2 Invoking and Servicing via Capabilities 68
7.3 Simple Client-Server Models 70
7.4 Resource Allocation 74
7.5 Semaphores Revisited 77
7.6 Data-Containing Semaphores 79
7.7 Shared Operations 80
7.8 Parameter Passing Details 83
Exercises 84

8. REMOTE PROCEDURE CALL 91
8.1 Mechanisms for Remote Procedure Call 91
8.2 Equivalence to Send/Receive Pairs 93

8.3 Return, Reply, and Forward Statements 96
Exercises 103

9. RENDEZVOUS 107
9.1 The Input Statement 108
9.1.1 General Form and Semantics 108
9.1.2 Simple Input Statements 109
9.2 Receive Statement Revisited 112
9.3 Synchronization Expressions 115
9.4 Scheduling Expressions 118
9.5 More Precise Semantics 119
9.6 Break And Continue Statements 120
9.7 Conditional Input 121
9.8 Arrays of Operations 122
9.9 Dynamic Operations 123
9.10 Return, Reply, and Forward Statements 124
Exercises 128

10. VIRTUAL MACHINES 139
10.1 Program Start-Up and Execution Overview 140
10.2 Creating Virtual Machines 141
10.3 Creating Remote Objects 143
10.4 Examples of Multiple Machine Programs 144
10.5 Predefined Fields 146
10.6 Parameterized Virtual Machines 149
10.7 Parameter Passing Details 151
10.8 Other Aspects of Virtual Machines 152
Exercises 153

11. THE DINING PHILOSOPHERS 159
11.1 Centralized Solution 160
11.2 Distributed Solution 162
11.3 Decentralized Solution 165
Exercises 169

12. EXCEPTIONS 173
12.1 Operations and Capabilities 173
12.2 Input Statements 174
12.3 Asynchronous Invocation 174
12.3.1 Handler Objects 175
12.3.2 Send 176
12.4 Additional Sources of Asynchrony 177
12.4.1 Exceptions After Reply 177
12.4.2 Exceptions After Forward 178
12.5 Exceptions and Operations 179
Exercises 180
13. INHERITANCE OF OPERATIONS 185
13.1 Operation Inheritance 186
13.2 Example: Distributing Operation Servicing 187
13.3 Example: Filtering Operation Servicing 188
13.4 Redefinition Considerations 190
Exercises 191
14. INTER-OPERATION INVOCATION SELECTION MECHANISM 193
14.1 Selection Method Expression 194
14.2 View Statement 197
14.2.1 General Form and Semantics 197
14.2.2 Simple View Statement 198
14.3 Selection Method Support Classes 198
14.3.1 ArmEnumeration Methods 199
14.3.2 InvocationEnumeration Methods 199
14.3.3 Invocation Methods 199
14.3.4 Timestamp Methods 199
14.4 Examples 200
14.4.1 Priority Scheduling 200
14.4.2 Random Scheduling 201
14.4.3 Median Scheduling 203
Exercises 204

Part II Applications

15. PARALLEL MATRIX MULTIPLICATION 211
- 15.1 Prescheduled Strips 212
- 15.2 Dynamic Scheduling: A Bag of Tasks 215
- 15.3 A Distributed Broadcast Algorithm 217
- 15.4 A Distributed Heartbeat Algorithm 220
- Exercises 223

16. SOLVING PDEs: GRID COMPUTATIONS 227
- 16.1 A Data Parallel Algorithm 228
- 16.2 Prescheduled Strips 232
- 16.3 A Distributed Heartbeat Algorithm 236
- 16.4 Using Multiple Virtual Machines 240
- Exercises 241

17. THE TRAVELING SALESMAN PROBLEM 247
- 17.1 Sequential Solution 248
- 17.2 Replicated Workers and a Bag of Tasks 251
- 17.3 Manager and Workers 254
- Exercises 258

18. A DISTRIBUTED FILE SYSTEM 263
- 18.1 System Structure 264
- 18.2 Directory and File Servers 266
- 18.3 User Interface 272
- Exercises 280

19. DISCRETE EVENT SIMULATION 283
- 19.1 A Simulation Problem 283
- 19.2 A Solution 285
  - 19.2.1 Main Class 285
  - 19.2.2 Processor Class 285
  - 19.2.3 Bus Controller Class 286
  - 19.2.4 Scheduler Class 288
- 19.3 Observations 290
- Exercises 291

20. INTERFACING JR AND GUIs 293
20.1 BnB Game Overview 293
20.2 BnB Code Overview 294
20.2.1 `Main` Class 296
20.2.2 `Window` Class 297
20.2.3 Button Class 299
20.2.4 `Board` Class 300
20.2.5 Toy Classes 305
20.2.6 Input Classes 307
20.3 Miscellany 308
Exercises 310
21. PREPROCESSORS FOR OTHER CONCURRENCY NOTATIONS 313
21.1 Conditional Critical Regions (CCRs) 313
21.2 Monitors 316
21.3 Communicating Sequential Processes (CSP) 320
Exercises 325
Appendices 331
A Synopsis of JR Extensions 331
B Invocation and Enumeration Classes 337
C Program Development and Execution 341
D Implementation and Performance 343
D.1 JR Virtual Machines 343
D.2 Remote Objects 344
D.2.1 Remote Class Loading 344
D.3 Operations and Operation Capabilities 345
D.4 Invocation Statements 345
D.4.1 Inheritance 346
D.5 Input Statements 346
D.6 Quiescence Detection 346
D.7 Performance Results 347
E History of JR 351
Bibliography 353

References 355

Index 359