Bernhard Beckert
Reiner Hähnle
Peter H. Schmitt

Verification of Object-Oriented Software

The KeY Approach

Foreword by K. Rustan M. Leino

Springer
Berlin Heidelberg New York
Hong Kong London
Milan Paris Tokyo

Contents

1 **Formal Methods for Software Construction**
by Reiner Hähnle 1
1.1 What KeY Is 1
1.2 About this Book 5
1.3 The Case for Formalisation 7
1.4 Creating Formal Requirements 10
1.5 Proof Obligations 14
1.6 Proving Correctness of Programs 15

Part I Foundations

2 **First-Order Logic**
by Martin Giese 21
2.1 Types 21
2.2 Signatures 25
2.3 Terms and Formulae 28
2.4 Semantics 31
2.4.1 Models 32
2.4.2 The Meaning of Terms and Formulae 35
2.4.3 Partial Models 40
2.5 A Calculus 44
2.5.1 An Example Proof 46
2.5.2 Ground Substitutions 47
2.5.3 Sequent Proofs 50
2.5.4 The Classical First-Order Rules 51
2.5.5 The Equality Rules 55
2.5.6 The Typing Rules 58
2.6 Soundness, Completeness 63
2.7 Incompleteness 65

3 Dynamic Logic
by Bernhard Beckert, Vladimir Klebanov, and Steffen Schlager . 69
3.1 Introduction . . . 69
3.2 Syntax . . . 71
3.2.1 Type Hierarchy and Signature . . . 71
3.2.2 Syntax of JAVA CARD DL Terms . . . 77
3.2.3 Syntax of JAVA CARD DL Updates . . . 77
3.2.4 Syntax of JAVA CARD DL Formulae . . . 80
3.3 Semantics . . . 87
3.3.1 Kripke Structures . . . 88
3.3.2 Semantics of JAVA CARD DL Updates . . . 92
3.3.3 Semantics of JAVA CARD DL Terms . . . 99
3.3.4 Semantics of JAVA CARD DL Formulae . . . 101
3.3.5 JAVA CARD-reachable States . . . 104
3.4 The Calculus for JAVA CARD DL . . . 108
3.4.1 Sequents, Rules, and Proofs . . . 108
3.4.2 Soundness and Completeness of the Calculus . . . 109
3.4.3 Rule Schemata and Schema Variables . . . 111
3.4.4 The Active Statement in a Modality . . . 114
3.4.5 The Essence of Symbolic Execution . . . 114
3.4.6 Components of the Calculus . . . 115
3.5 Calculus Component 1: Non-program Rules . . . 116
3.5.1 First-order Rules . . . 116
3.5.2 The Cut Rule and Lemma Introduction . . . 118
3.5.3 Non-program Rules for Modalities . . . 119
3.6 Calculus Component 2: Reducing JAVA Programs . . . 119
3.6.1 The Basic Assignment Rule . . . 119
3.6.2 Rules for Handling General Assignments . . . 120
3.6.3 Rules for Conditionals . . . 124
3.6.4 Unwinding Loops . . . 125
3.6.5 Replacing Method Calls by their Implementation . . . 126
3.6.6 Instance Creation and Initialisation . . . 135
3.6.7 Handling Abrupt Termination . . . 142
3.7 Calculus Component 3: Invariant Rules for Loops . . . 146
3.7.1 The Classical Invariant Rule . . . 146
3.7.2 Loop Invariants and Abrupt Termination in JAVA CARD DL . . . 147
3.7.3 Implementation of Invariant Rules . . . 151
3.7.4 An Improved Loop Invariant Rule . . . 153
3.8 Calculus Component 4: Using Method Contracts . . . 162
3.9 Calculus Component 5: Update Simplification . . . 166
3.9.1 General Simplification Laws . . . 167
3.9.2 Update Normalisation . . . 168
3.9.3 Update Application . . . 172
3.10 Related Work . . . 175

4 **Construction of Proofs**
by Philipp Rümmer 177
4.1 Taclets by Example 181
4.2 Schema Variables 190
4.2.1 The Kinds of Schema Variables in Detail 191
4.2.2 Schematic Expressions 195
4.2.3 Instantiation of Schema Variables and Expressions 196
4.2.4 Substitutions Revisited 198
4.2.5 Schema Variable Modifiers 201
4.2.6 Schema Variable Conditions 201
4.2.7 Generic Types 202
4.2.8 Meta-Operators 206
4.3 Instantiations and Meta Variables 207
4.4 Systematic Introduction of Taclets 209
4.4.1 The Taclet Language 209
4.4.2 Managing Rules: An Excursion to Taclet Options 214
4.4.3 Well-Formedness Conditions on Taclets 216
4.4.4 Implicit Bound Renaming and Avoidance of Collisions 218
4.4.5 Applicability of Taclets 220
4.4.6 The Effect of a Taclet 224
4.4.7 Taclets in Context: Taclet-Based Proofs 225
4.5 Reasoning about the Soundness of Taclets 227
4.5.1 Soundness in Sequent Calculi 229
4.5.2 A Basic Version of Meaning Formulae 229
4.5.3 Meaning Formulae for Rewriting Taclets 232
4.5.4 Meaning Formulae in the Presence of State Conditions 233
4.5.5 Meaning Formulae for Nested Taclets 235
4.5.6 Elimination of Schema Variables 237
4.5.7 Introducing Lemmas in KeY 239

Part II Expressing and Formalising Requirements

5 **Formal Specification**
by Andreas Roth and Peter H. Schmitt 243
5.1 General Concepts 243
5.1.1 Operation Contracts 244
5.1.2 Invariants 246
5.2 Object Constraint Language 248
5.2.1 OCL by Example 248
5.2.2 OCL Syntax 254
5.2.3 OCL Semantics 263
5.2.4 Advanced Topics 269
5.3 Java Modeling Language 275
5.3.1 JML by Example 276

5.3.2 JML Expressions 280
5.3.3 Operation Contracts in JML 282
5.3.4 Invariants in JML 285
5.3.5 Model Fields and Model Methods 286
5.3.6 Supporting Verification with Annotations 289
5.4 Comparing OCL and JML 290

6 Pattern-Driven Formal Specification
by Richard Bubel and Reiner Hähnle 293
6.1 Introduction 293
6.2 The *Database Query* Specification Pattern 294
6.2.1 Relational Database Query 295
6.2.2 Pattern Usage Example 302
6.3 Specification Patterns 304
6.3.1 Format of Specification Patterns 304
6.3.2 Application of Specification Patterns 305
6.3.3 Other Pattern Usage Scenarios 305
6.4 Simplification of Pattern-Generated Constraints 306
6.5 Support for Specification Patterns in KeY 308
6.6 Conclusion and Future Work 311

7 Natural Language Specifications
by Kristofer Johannisson 315
7.1 Feature Overview 315
7.1.1 Translating OCL to Natural Language 315
7.1.2 Multilingual Specification Editor 316
7.1.3 Suggested Use Cases 319
7.2 The Grammatical Framework 321
7.2.1 GF Examples 322
7.3 System Overview 323
7.4 The Multilingual Editor 325
7.4.1 Syntax-Directed Editing 325
7.4.2 Top-Down Editing: Refinement 325
7.4.3 Bottom-Up Editing: Wrapping 325
7.4.4 Other Editor Features 326
7.4.5 Expressions and Sentences 328
7.4.6 Subtyping 328
7.5 Translation of Domain Specific Concepts 328
7.5.1 Grammar Generation 329
7.5.2 Customising the Translation 329
7.6 Further Reading 330
7.7 Summary 330

8 **Proof Obligations**
by Andreas Roth 331
8.1 Design Validation 333
8.1.1 Disjoint Preconditions 333
8.1.2 Behavioural Subtyping of Invariants 334
8.1.3 Behavioural Subtyping of Operations 335
8.1.4 Strong Operation Contract 338
8.2 Observed-State Correctness 340
8.2.1 Observed States vs. Visible States 341
8.2.2 Assumptions before Operation Calls 344
8.2.3 Operation Calls 345
8.2.4 Assertions After Operation Calls 347
8.2.5 Static Initialisation 349
8.3 Lightweight Program Correctness 351
8.3.1 Invariants 351
8.3.2 Postconditions and Termination 352
8.3.3 Modifies Clauses 352
8.4 Proving Entire Correctness 355
8.5 Modular Verification 359
8.5.1 Visibility-Based Approach 359
8.5.2 Encapsulation-Based Approach 361
8.5.3 Verification Strategies 366
8.5.4 Components and Modular Proofs 368

9 **From Sequential Java to Java Card**
by Wojciech Mostowski 371
9.1 Introduction 371
9.2 Motivation 372
9.3 Java Card Memory, Atomicity, and Transactions 373
9.4 Strong Invariants: The “Throughout” Modality 375
9.4.1 Additional Calculus Rules for “Throughout” 376
9.5 Handling Transactions in the Logic 378
9.5.1 Rules for Beginning and Ending a Transaction 378
9.5.2 Rules for Conditional Assignment 382
9.6 Examples 383
9.7 Non-atomic Java Card API Methods 388
9.7.1 Transaction Suspending and Resuming 390
9.7.2 Conditional Assignments Revised 392
9.8 Summary 394
9.8.1 Related Work 394
9.9 Implementation of the Rules 395
9.9.1 New Modalities 395
9.9.2 Transaction Statements and Special Methods 395
9.9.3 Taclet Options 398
9.9.4 Implicit Fields 398

9.9.5 Conditional Assignment Rule Taclets . . . 399
9.9.6 Examples in the KeY System . . . 400
9.9.7 Current Limitations . . . 401

Part III Using the KeY System

10 Using KeY
by Wolfgang Ahrendt . . . 405
10.1 Introduction . . . 405
10.2 Exploring Framework and System Simultaneously . . . 408
10.2.1 Exploring Basic Notions And Usage . . . 408
10.2.2 Exploring Terms, Quantification, and Instantiation . . . 420
10.2.3 Exploring Programs in Formulae . . . 428
10.3 Generating Proof Obligations . . . 443

11 Proving by Induction
by Angela Wallenburg . . . 449
11.1 Introduction . . . 449
11.2 The Need for Induction . . . 449
11.2.1 A First Look at an Induction Rule . . . 450
11.2.2 A Small Example . . . 450
11.3 Basics of Induction in KeY . . . 452
11.3.1 Induction Rule . . . 452
11.3.2 Induction Variable . . . 453
11.3.3 Induction Formula . . . 453
11.3.4 Induction Principle . . . 453
11.4 A Simple Program Loop Example . . . 454
11.4.1 Preparing the Proof . . . 455
11.4.2 The Proof in JAVA CARD DL . . . 455
11.4.3 Making the Proof in the KeY System . . . 459
11.5 Choosing the Induction Variable . . . 460
11.5.1 The Difficulty of Guiding Induction Proofs . . . 460
11.5.2 How to Choose the Induction Variable . . . 460
11.6 Different Induction Rules . . . 463
11.6.1 Customised Induction Rules . . . 464
11.6.2 The Noetherian Induction Rule . . . 467
11.6.3 Soundness of Induction Rules . . . 468
11.7 Generalisation of Induction Formulae . . . 469
11.7.1 Cubic Sum Example . . . 469
11.8 Summary: The Induction Proving Process . . . 472
11.9 Conclusion . . . 474

12 Java Integers
by Steffen Schlager . . . 475
12.1 Motivation . . . 475
12.2 Integer Types in Java . . . 477
12.2.1 Implicit Type Casts . . . 479
12.2.2 Differences between Java and Java Card . . . 480
12.3 Refinement and Retrenchment . . . 481
12.3.1 Preliminaries . . . 481
12.3.2 Refinement . . . 482
12.3.3 Retrenchment . . . 483
12.4 Retrenching Integers in KeY . . . 486
12.4.1 Weakening the Postcondition . . . 487
12.4.2 Strengthening the Precondition . . . 489
12.5 Implementation . . . 491
12.5.1 Sequent Calculus Rules . . . 492
12.5.2 Example . . . 494
12.6 Pitfalls Related to Integers . . . 496
12.7 Conclusion . . . 497
12.8 Related Work . . . 497

13 Proof Reuse
by Vladimir Klebanov . . . 499
13.1 Introduction . . . 499
13.2 A Running Example . . . 500
13.3 The Main Reuse Algorithm . . . 502
13.4 Computing Rule Application Similarity . . . 505
13.5 Finding Reusable Subproofs . . . 510
13.6 Implementation and a Short Practical Guide . . . 512
13.7 The Example Revisited . . . 512
13.8 Other Systems and Related Methods . . . 514
13.9 Reuse as a Proof Search Framework . . . 516
13.10 Conclusion . . . 519

Part IV Case Studies

14 The Demoney Case Study
by Wojciech Mostowski . . . 523
14.1 Introduction . . . 523
14.2 Demoney . . . 524
14.3 OCL, JML, and Dynamic Logic . . . 524
14.4 Modular Verification . . . 527
14.4.1 KeY Built-in Methods . . . 528
14.5 Properties . . . 529
14.5.1 Functional Properties . . . 529

14.5.2 Security Properties 533
14.5.3 Only ISOExceptions at Top Level 535
14.5.4 Atomicity and Transactions 547
14.5.5 No Unwanted Overflow 552
14.5.6 Other Properties 553
14.6 Lessons 555
14.6.1 Related Work 556

15 The Schorr-Waite-Algorithm
by Richard Bubel 559
15.1 The Algorithm in Detail 559
15.1.1 In Theory 559
15.1.2 In Practice 561
15.2 Specifying Schorr-Waite 563
15.2.1 Specifying Reachability Properties 564
15.2.2 Specification in JAVA CARD DL 568
15.3 Verification of Schorr-Waite within KeY 572
15.3.1 Replacing Arguments of Non-Rigid Functions behind Updates 573
15.3.2 The Proof 574
15.4 Related Work 576

Appendices

A Predefined Operators in JAVA CARD DL
by Steffen Schlager 581
A.1 Syntax 581
A.1.1 Built-in Rigid Function Symbols 581
A.1.2 Built-in Rigid Function Symbols whose Semantics Depends on the Chosen Integer Semantics 582
A.1.3 Built-in Non-Rigid Function Symbols 583
A.1.4 Built-in Rigid Predicate Symbols 584
A.1.5 Built-in Rigid Predicate Symbols whose Semantics Depends on the Chosen Integer Semantics 585
A.1.6 Built-in Non-rigid Predicate Symbols 585
A.2 Semantics 585
A.2.1 Semantics of Built-in Rigid Function Symbols 585
A.2.2 Semantics of Built-in Predicate Symbols 587

B The KeY Syntax
by Wojciech Mostowski 589
B.1 Notation, Keywords, Identifiers, Numbers, Strings 590
B.2 Terms and Formulae 592
B.2.1 Logic Operators 592

B.2.2 Atomic Terms 595
B.3 Rule Files 602
B.3.1 Library and File Inclusion 602
B.3.2 Rule File Declarations 603
B.3.3 Rules 607
B.4 User Problem and Proof Files 609
B.4.1 Method Contracts 611
B.5 Schematic JAVA Syntax 613
B.5.1 Method Calls, Method Bodies, Method Frames 613
B.5.2 Exception Catching in Contracts 614
B.5.3 Inactive JAVA Block Prefix and Suffix 614
B.5.4 Program Schema Variables 615
B.5.5 Meta-Constructs 615
B.5.6 Passive Access in Static Initialisation 616

References 617

List of Symbols 635

Index 639