Universiteit Leiden
The Netherlands

Reasoning about object-oriented programs: from classes to interfaces

Bian, J.

Citation

Bian, J. (2024, May 21). *Reasoning about object-oriented programs: from classes to interfaces*. Retrieved from https://hdl.handle.net/1887/3754248

Version:	Publisher's Version
License:	Licence agreement concerning inclusion of doctoral thesis in the Institutional Repository of the University of Leiden
Downloaded from:	https://hdl.handle.net/1887/3754248

Note: To cite this publication please use the final published version (if applicable).

Reasoning about object-oriented programs: from classes to interfaces

Jinting Bian

Copyright © 2024 Jinting Bian, All Rights Reserved

ISBN 978-94-6496-122-5

Cover design: Zhiyu Gu

Reasoning about object-oriented programs: from classes to interfaces

Proefschrift

ter verkrijging van
de graad van doctor aan de Universiteit Leiden,
op gezag van rector magnificus prof.dr.ir. H. Bijl,
volgens besluit van het college voor promoties
te verdedigen op dinsdag 21 mei 2024
klokke 11:15 uur

door Jinting Bian

geboren te Taiyuan,China

in 1994

Promotores: Prof. dr. F.S. de Boer
Prof. dr. M.M. Bonsangue

Promotiecommissie: Prof. dr. R.V. van Nieuwpoort
Prof. dr. H.C.M. Kleijn
Prof. dr. M.-C. Jakobs (Ludwig-Maximilians Universität)
Prof. dr. M. Sirjani (Malardalen University)
Dr. E. Poll (Radboud Universiteit)
Dr. A.W. Laarman

Jinting Bian was financially supported through the China Scholarship Council (CSC) to participate in the PhD programme of Leiden University. Grant number 202007720094.

The research in this thesis was performed at the Center for Mathematics and Computer Science (CWI) in Amsterdam and Leiden Institute of Advanced Computer Science at Leiden University, under the auspices of the research school IPA (Institute for Programming research and Algorithmics).

Copyright © 2024 Jinting Bian.

Contents

1 Introduction **1**
1.1 Background . . . 1
1.2 Thesis outline . . . 4
1.3 Main challenges and contributions . . . 5

2 Preliminaries **11**
2.1 Object-oriented programming . . . 11
2.2 Java Collection Framework . . . 12
2.3 Theorem prover overview . . . 14
2.3.1 KeY theorem prover . . . 14
2.3.2 Isabelle/HOL theorem prover . . . 17

3 Class specification and verification: a step-by-step guide **21**
3.1 Introduction . . . 22
3.2 Preliminaries . . . 23
3.3 Structure and behavior of `java.util.LinkedList` . . . 25
3.3.1 Expected and unexpected method behavior . . . 28
3.3.2 Integer overflow bug . . . 29
3.3.3 Verification goal . . . 30
3.4 Formulating a class invariant . . . 32
3.5 The acyclicity property . . . 35
3.6 The `add` method . . . 35
3.7 The `remove` method . . . 38
3.8 Summary . . . 44

4 History-based reasoning about interfaces **47**
4.1 Introduction . . . 48
4.2 Background . . . 49
4.3 Specification and verification challenges for the `Collection` interface 50
4.4 History-based reasoning approach . . . 54
4.4.1 The executable history-based approach . . . 54
4.4.2 The logical history-based approach . . . 55
4.4.3 Comparative analysis . . . 56
4.5 Summary . . . 58

5 Executable history-based reasoning: a case study **59**
5.1 Introduction . . . 60
5.2 History-based specification in KeY . . . 60
5.2.1 The `History` class for `Collection` . . . 60
5.2.2 Attributes of `History` . . . 62
5.2.3 The `Collection` interface . . . 65
5.2.4 History-based refinement . . . 66
5.3 History-based verification of `Collection` . . . 67
5.4 Summary . . . 71

6 Logical history-based reasoning: an advanced case study **73**
6.1 Introduction . . . 74
6.2 Intergrating Abstract Data Types in KeY . . . 74
6.3 History-based specification . . . 79
6.3.1 History abstractions . . . 82
6.3.2 Method contracts of `Collection` . . . 85
6.4 History-based client-side verification . . . 90
6.4.1 Significant improvement in proof effort . . . 91
6.4.2 Reasoning about `Iterator` . . . 93
6.4.3 Reasoning about binary methods . . . 95
6.4.4 Proof statistics . . . 98
6.5 Summary . . . 99

7 History-based reasoning about behavioral subtyping **101**
7.1 Introduction . . . 102
7.2 Methodology . . . 103
7.2.1 History-based refinement theory . . . 105
7.2.2 Verifying method call and method implementation . . . 106
7.3 Case study . . . 107
7.3.1 History-based reasoning . . . 108
7.3.2 History-based specification . . . 111
7.3.3 Behavioral subtyping . . . 113
7.3.4 Example of method call and implementation . . . 116
7.4 Summary . . . 117

8 Conclusion **119**
8.1 Summary of contributions . . . 119
8.2 Future work . . . 121

Bibliography 125

English summary 131

Nederlandse samenvatting 133

Acknowledgements 135

Curriculum Vitae 137

Publication List 139