



Universiteit
Leiden
The Netherlands

Verifying OCL specifications of UML models : tool support and compositionality

Kyas, M.

Citation

Kyas, M. (2006, April 4). *Verifying OCL specifications of UML models : tool support and compositionality*. Lehmanns Media. Retrieved from <https://hdl.handle.net/1887/4362>

Version: Corrected 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/1887/4362>

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

Lehmanns Media **LOB.de**

Verifying OCL Specifications of UML Models: Tool Support and Compositionality

PROEFSCHRIFT

ter verkrijging van
de graad van Doctor aan de Universiteit Leiden,
op gezag van de Rector Magnificus Dr. D. D. Breimer,
hoogleraar in der Faculteit der Wiskunde en
Natuurwetenschappen en die der Geneeskunde,
volgens besluit van het College voor Promoties
te verdedigen op dinsdag 4 april 2006
te klokke 14.15 uur

door

Marcel Kyas
geboren te Pinneberg, Duitsland
in 1975

Promotiecommissie

Promotores: Prof. dr. J.N. Kok
Prof. dr. W.-P. de Roever
Christian-Albrechts-Universität zu Kiel

Copromotor: Dr. F.S. de Boer

Referent: Prof. dr. Olaf Owe
Universitetet i Oslo

Overige leden: Prof. dr. F. Arbab
Prof. dr. G. Rozenberg
Prof. dr. S.M. Verduyn Lunel

Part of this work has been financially supported by IST project OMEGA (IST-33522-2001) and NWO/DFG project Mobi-j (RO 1122/9-1, RO 1122/9-2). The work has been carried out at the Christian-Albrechts-Universität zu Kiel, Germany.



The work in this thesis has been carried out under the auspices of the research school IPA (Institute for Programming research and Algorithmics).

Bibliografische Informationen der Deutschen Bibliothek:

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.ddb.de/> abrufbar.

Verifying OCL Specifications of UML Models: Tool Support and Compositionality / Marcel Kyas. – With ref.

Berlin: Lehmanns Media, LOB.de, 2006.

Zugl.: Dissertation, Universiteit Leiden.

Zugl.: IPA Dissertation Series 2006-05.

ISBN: 3-86541-142-8

Printed by docupoint GmbH, Magdeburg, Germany.

Cover photograph "Dog sledging on Svalbard," © 2005 by Martin Steffen. Used with permission.

© 2005, Marcel Kyas. All rights reserved.

Preface

Another year is gone;
and I still wear
straw hat and straw sandal.

(Bashō)

This dissertation describes the results of my research at the Chair of Software Technology at the Christian-Albrechts-Universität zu Kiel, which was conducted as part of the Omega project (IST-2001-33522, see also <http://www-omega.imag.fr/>). The aim of Omega was to create a development method in UML (Unified Modelling Language) for embedded and real-time systems built on formal footing. My task was to adapt OCL (Object Constraint Language) to the Omega method. My solution is to extend OCL to a trace-based specification language facilitating compositional reasoning. Especially, I had to solve problems related to object-orientation and object creation.

OCL requires a context in which it is interpreted. This context is provided by other UML diagrams. However, UML is a large language. For this dissertation I have selected class-diagrams and state-machines for providing the context for OCL.

Class diagrams describe the structure of a system. They are well-understood and a very stable part of UML. They were present in the beginning of UML in 1997 and are based on entity-relationship diagrams, introduced in 1976 [24]. Considering this almost thirty year history, class diagrams will probably not change drastically anymore.

State machines provide a notation for describing the behaviour of systems. They evolved from Harel's statecharts [60] and their object-oriented development [61]. According to Hewitt [63], event-driven semantics are the essence of object-oriented computation. Such a semantics is used for state machines and allowed me to avoid all complications of more "modern" object-oriented programming languages.

While I was researching for my dissertation, the UML 2.0 standard has been actively developed. Many promising ideas have been introduced. It was a considerable amount of work to follow these developments and often results became obsolete or invalid. I hope that tracking this moving target contributed to making my results more robust.

Acknowledgements. I thank all people who supported me during the time of researching and writing this thesis.

Bengt Jonsson has been a very insightful reviewer of the Omega project. His opinion

Preface

on my deliverables has helped to improve this dissertation. He generously took the time to explain his work and views on trace-based specification and verification of systems.

Willem-Paul de Roever was very attentive of my well-being, sometimes “commanding” me to take a holiday. When I followed his “order” in 2001, I decided to visit Oslo, because at that time my fiancée was studying there. Willem-Paul suggested that I visit Ole-Johan Dahl on this occasion. Instead, I met Olaf Owe who told me that Ole-Johan Dahl was very ill. In spite of me not being invited he took the time to introduce me to his research. I was impressed by his work and suggested that he would present his work in Kiel. Instead he sent Einar Broch Johnsen. This set up the foundation for fruitful collaborations between Oslo, Kiel, and Frank de Boer in Amsterdam. Einar’s most memorable contribution was to organise a working meeting on Spitsbergen during February 2005, as displayed on this book’s cover.

Frank de Boer proved to be an excellent troubleshooter when we wrote a paper together. He always came up with new ideas whenever a problem arose *and* found the time to discuss them with me.

Martin Steffen’s encyclopedic knowledge — not limited to computer science — his attentive observations, and his questions helped me to understand my own ideas better.

I also thank my numerous co-authors: Harald Fecher and Jens Schönborn discussed the subtleties of the semantics of UML; especially Jens worked out the semantics of UML 2.0 state machines, which helped me to understand the version used in the Omega project. Mark van der Zwaag formalised the semantics of Omega state machines in PVS and explained it to me. I based the translation of OCL into PVS on his work. Hillel Kugler and Tamarah Arons were the testers of this translator. Without their feedback, patience, and willingness to work with an evolving tool, many problems with the theory and its implementations would have emerged later, if at all. Jozef Hooman has been a great help in working with PVS and modelling the MARS case study.

Mirco Kuhlmann volunteered to read various drafts of this dissertation. He has pointed out many mistakes and omissions and suggested many improvements.

Our secretaries Sabine Hilge and especially Anne Straßner handled most bureaucratic tasks reliably. Thanks to their effort I was able to focus on research and teaching.

I thank my friends for reminding me of a life with leisure. Especially Jan Engelbach and Ortwin Ebhardt invited me for a chat over lunch or a party whenever the occasion arose.

I thank my parents Heidemarie and Horst for their loving encouragement and support. I shall always remain grateful for their advice.

Finally, my fiancée Ann-Dörte took care of me and most of the daily errands during the final stages in writing this thesis, reminded me to take shorter and longer breaks from work, and was wonderfully supportive and loving. She deserves all the praise for the fact that I remained healthy and sane in this memorable part of my life.

Marcel Kyas
January 26, 2006, Kiel

Contents

Preface	i
1 Introduction	1
1.1 Unified Modelling Language and Object Constraint Language	3
1.2 Problem Statement	6
1.2.1 Correctness of Systems	7
1.2.2 Compositionality	9
1.3 Contribution of this Dissertation	10
1.4 Publication History	11
1.5 Dissertation Outline	12
2 Introduction to Models of OCL and UML	13
2.1 Class Diagrams	13
2.1.1 Generalisation	16
2.1.2 Association	18
2.1.3 Parameterisation	19
2.2 Object Diagrams	19
2.2.1 Associations and Navigation Expressions	20
2.2.2 Relating Object Diagrams to Class Diagrams	24
2.3 Object Constraint Language	26
2.3.1 Context of Constraints	26
2.3.2 Abstract Syntax	27
2.3.3 Semantics	30
2.3.4 OCL Standard Library	40
2.3.5 Critique of OCL	50
2.4 State Machines	55
2.5 Summary	57
3 Type Checking OCL	59
3.1 Introduction	59
3.2 State of the Art	60
3.3 Extensions	66
3.3.1 Intersection Types	66
3.3.2 Union Types	69

Contents

3.3.3	Parametric Polymorphism	71
3.3.4	Bounded Operator Abstraction	72
3.3.5	Flattening and Accessing the Run-Time Type of Objects . . .	74
3.4	Adequacy and Decidability	75
3.5	Related Work and Conclusions	77
4	Formalising UML Models and OCL Constraints in PVS	79
4.1	Introduction	79
4.2	Shallow versus Deep Embedding	80
4.3	PVS Language	82
4.4	Running Example	83
4.5	Definition of the Translator	85
4.5.1	Front End	85
4.5.2	Middle End	87
4.5.3	Back End	90
4.6	Soundness of the Translation	100
4.7	Summary and Conclusion	108
5	Trace-based Compositional Specification and Verification	109
5.1	Introduction	109
5.2	State of the Art and Motivation	111
5.3	Observables	113
5.3.1	Events	113
5.3.2	History	117
5.3.3	Comparison to OCL 2.0	117
5.4	Local and Global Specifications	120
5.4.1	Local Specification Language	121
5.4.2	Global Specification Language	123
5.5	Compatibility	124
5.6	Conclusions, Related Work, and Future Work	126
6	A Compositional Trace Logic for Behavioural Interface Specifications	129
6.1	Introduction	129
6.2	Interfaces	130
6.3	Trace Logic	130
6.4	Compositionality	132
6.5	Axiomatisation	133
6.5.1	Observing Object Creation	133
6.5.2	Communication Mechanisms	140
6.6	Sieve of Eratosthenes	141
6.7	Related Work	144
6.8	Conclusion and Future Work	145

7	Compositional Verification of Timed Components in PVS	147
7.1	Introduction	147
7.2	Semantics	148
7.3	Compositional Proof Rules	150
7.4	The MARS Example	150
7.5	Decomposition of the MARS example	154
7.5.1	Message Receiver	154
7.5.2	Error Logic	157
7.6	Correctness of the decomposition	160
7.7	Conclusions	160
8	Conclusion	163
8.1	Summary	163
8.2	Future Research	164
A	OCL 2.0 Grammar	167
A.1	Literals	167
A.2	Files	168
A.3	Expressions	168
B	Semantics of OCL in PVS	171
	Bibliography	181
	Summary	195
	Samenvatting	197
	Curriculum Vitæ	199

Contents

List of Tables

2.1	Example valuations of associations	22
2.2	Semantics of boolean connectives	43
3.1	Kinding system	61
3.2	Definition of type conformance	63
3.3	Typing rules for OCL	64
3.4	Typing rules for OCL (continued)	65
3.5	Intersection types	69
3.6	Rules for union types	70
3.7	Subtyping rules for parametric polymorphism	72
A.1	Reserved keywords	167
B.1	Mapping OCL types to PVS types	171
B.2	Representing Set operations in PVS	177
B.3	Representing Sequence operations in PVS	178
B.4	Representing Bags operations in PVS	179

List of Tables

List of Figures

2.1	Valid and invalid generalisations	17
2.2	Valid and invalid associations	18
2.3	Navigation example	22
2.4	State machine	56
3.1	A simple initial class diagram	67
3.2	The same diagram after a change	67
3.3	A simple example class diagram	70
4.1	Class diagram of the Sieve example	84
4.2	State machine of the Generator	84
4.3	State machine of a Sieve	84
4.4	Architecture of the translator	85
4.5	Example class diagram	87
4.6	Translation of the Sieve class diagram	91
4.7	Translation of the Generator state machine	93
5.1	Definition of a communication record	114
5.2	Properties of histories	118
5.3	Properties of OCL 2.0's <i>OclMessage</i>	118
7.1	Architecture of the data bus manager	151
7.2	Data with period P and jitter J	151
7.3	State machine of the data source	152
7.4	Decomposed architecture for two data sources	154
7.5	State machine of the message receiver	155
7.6	State Machine of the error logic component	157

List of Figures