



Francisco Dantas de Medeiros Neto

**On the Role of Composition Properties on
Program Stability**

TESE DE DOUTORADO

Thesis presented to the Programa de Pós-Graduação em
Informática of the Departamento de Informática, PUC-Rio as
partial fulfillment of the requirements for the degree of Doutor
em Informática

Advisor: Prof. Alessandro Fabricio Garcia

Rio de Janeiro
March, 2013



Francisco Dantas de Medeiros Neto

**On the Role of Composition Properties on
Program Stability**

Thesis presented to the Programa de Pós-Graduação em Informática, of the Departamento de Informática do Centro Técnico Científico da PUC-Rio, as partial fulfillment of the requirements for the degree of Doutor.

Prof. Alessandro Fabricio Garcia

Advisor

Departamento de Informática — PUC-Rio

Prof. Silvia Mara Abrahao

Faculty of Computer Science

Universidad Politecnica de Valencia

Prof. Rosana Teresinha Vaccare Braga

Instituto de Ciências Matemáticas e Computação

Universidade de São Paulo

Prof. Carlos José Pereira de Lucena

Departamento de Informática - PUC-Rio

Prof. Simone Diniz Junqueira Barbosa

Departamento de Informática - PUC-Rio

Prof. José Eugenio Leal

Coordinator of the Centro Técnico Científico da PUC-Rio

Rio de Janeiro, March 15th, 2013

All rights reserved. Copying portions or the entirety of the work is prohibited, except as otherwise permitted by the university, the author, and the supervisors.

Francisco Dantas de Medeiros Neto

Francisco Dantas joined PUC-Rio as a PhD student on Software Engineering at the Informatics Department in 2009 under Dr. Alessandro Garcia's supervision. He visited the University of Lancaster in the UK from 2011 to 2012, where he worked with Professor Jon Whittle's software engineering team. During his PhD he published 18 papers in renowned workshops and conferences, such as ICSE, ESEM and AOSD. He was also a collaborator in under-graduation and post-graduation courses at PUC-Rio. In addition, he has also been a collaborator and has participated in the writing of a number of research projects. Francisco began his history in Computer Science in 1997, when he started his Computer Science course at the Federal University of Rio Grande do Norte (UFRN), Brazil. He received his degree in Computer Science from the UFRN in 2001 and the M.Sc. Degree in Computer Science from UFRN as well in 2004. He worked as a substitute teacher at the Informatics Department (UFRN) for three years and in 2005 he joined the State University of Rio Grande do Norte as an Assistant Professor, where he is working up to now. His main research interests are Advanced Techniques for Modular Programming, Product Lines, Software Metrics, Empirical Software Engineering and Software Architecture areas.

Ficha Catalográfica

Medeiros Neto, Francisco Dantas de

On the role of composition properties on program stability / Francisco Dantas de Medeiros Neto ; advisor: Alessandro Fabricio Garcia. – 2013.

166 f. ; 30 cm

Tese (doutorado)–Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2013.

Inclui bibliografia

1. Informática – Teses. 2. Propriedades de composição. 3. Mecanismos de composição. 4. Estabilidade de programas. 5. Métricas de software. I. Garcia, Alessandro Fabricio. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

*To **my parents** in memory.
To **Mirian** for the family support.
To **Camila** for her affection.
To **Thais Batista** for believing in me and putting
me back into the academic life.*

Acknowledgments

Thank you God

*For the people you have put in my life
For the support that I have had all the time
For answering my prayers every time*

Thank you Alessandro

*For guiding me on this journey
For feeding my academic soul
For walking towards the same goal*

Thank you Jon Whittle

*For the collaboration
For the friendship
For the long and great conversations*

Thank you Examiners

*For accepting to be here today
For evaluating my research
For coming from far away*

Thank you Mirian

*For being my sister
For never disappearing
For you big heart, my darling*

Thank you Camila

*For walking with me since the beginning
For your affection and patience
For supporting me to realize my dream*

Thank you Isela

*For being with me
For your laugh
For always listening to me*

Thank you Vera

*For your eternal youth
For your heart and doses of laughter
For your phone booth*

Thank you Thais Batista

For giving me back the academic life

For your constant support

For your big heart

Thank you my OPUS group of friends

For having made my 1460 days

For the doses of work and fun

For understanding my run

Thank you my special big friends

Ingrid for your companionship

Rachel for your patience and sweetness

Rosi for your constant support and friendship

Thank you DI collaborators

Regina, Ruth, Tereza, Alex and Wagner

For helping during this time

For being available all the time

Thank you CNPq and CAPES

From the depths of my heart

For the financial support

Abstract

Dantas, Francisco; Garcia, Alessandro. **On the Role of Composition Properties on Program Stability**. Rio de Janeiro, 2013. 166p. DSc Thesis — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

The demand for incremental software development has driven a search for advanced programming techniques, such as aspect-oriented programming and feature-oriented programming. These techniques share the goal of supporting localized implementation of software changes in order to promote program stability. To achieve this goal, they offer a wide range of sophisticated composition mechanisms, which provide means to flexibly define the composition of two or more modules in a program. However, given the complexity of the resulting composition code, the initial definition and further changes to a single composition specification might affect the structure and behaviour of multiple modules, thereby harming the program stability. A complicating factor is that those changes often require some reasoning about certain composition properties, which are not explicit in the implementation or design artefacts. Unfortunately, there is no understanding in the state of the art about the composition properties that affect positively or negatively the program stability. This understanding is not yet possible as: (i) there is no conceptual characterization and quantification means for composition code properties, and (ii) there is no empirical investigation on the influence of these properties on program stability. A side effect of these gaps is that developers have resorted to conventional metrics, such as coupling, to determine or predict the stability of a program implemented with advanced programming techniques. In this context, this thesis presents three contributions to overcome the aforementioned problems. First, we have developed an empirical study revealing that widely-used metrics, such as coupling, are not effective indicators of stability when advanced programming techniques are used. Second, we propose a measurement framework encompassing a suite of composition metrics intended to quantify properties of the composition code. This framework is based on a meta-model and terminology for characterizing the elements and properties of the composition code. This framework is extensible and agnostic to particular programming techniques. Third, we also investigate how to alleviate the maintenance effort in performing changes related to the composition code. We evaluate if the availability of design models enriched with

specification of composition properties help developers to improve program stability in their maintenance tasks.

Keywords

Composition Properties. Composition Mechanisms. Program Stability. Software Metrics.

Resumo

Dantas, Francisco; Garcia, Alessandro. **Análise de Propriedades de Código de Composição em Estabilidade de Programas**. Rio de Janeiro, 2013. 166p. Tese de Doutorado — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A demanda por desenvolvimento de software incremental tem impulsionado a busca por técnicas de programação avançadas, tais como programação orientada a aspectos e programação orientada a características. Estas técnicas têm por objetivo apoiar a implementação de mudanças de software de forma localizada, a fim de promover a estabilidade do programa. Para atingir este objetivo, elas oferecem uma grande variedade de sofisticados mecanismos de composição, que fornecem meios para definir de forma flexível a composição de dois ou mais módulos de um programa. No entanto, dada a complexidade do código composição resultante, a definição inicial e alterações posteriores na especificação de uma simples composição podem afetar a estrutura e o comportamento de vários módulos, prejudicando assim a estabilidade do programa. Um fator complicador é que essas mudanças geralmente exigem raciocínio sobre certas propriedades da composição, que não estão explícitas nos artefatos de implementação ou de projeto. Infelizmente, não há conhecimento do estado da arte sobre as propriedades da composição que afetam positivamente ou negativamente a estabilidade do programa. Esse entendimento não é possível ainda, uma vez que: (i) não há uma caracterização conceitual e meios de quantificação referentes às propriedades do código de composição, e (ii) não há nenhuma investigação empírica sobre a influência dessas propriedades na estabilidade do programa. Um efeito colateral dessas lacunas é que os desenvolvedores têm recorrido a métricas convencionais, tais como o acoplamento, para determinar ou prever a estabilidade de um programa implementado usando técnicas de programação avançadas. Neste contexto, a presente tese apresenta três contribuições. Primeiro, são apresentados os resultados de um estudo empírico, revelando que as métricas convencionais utilizadas, tais como acoplamento, não são indicadores eficazes de estabilidade quando técnicas avançadas de programação são usadas. Em segundo lugar, é apresentado um arcabouço de medição que engloba um conjunto de métricas de composição destinado a quantificar as propriedades do código de composição. Este arcabouço foi desenvolvido com base em uma meta-modelo e uma terminologia usada para caracterizar os elementos e propriedades do código de composição. Trata-se de um arcabouço extensível e que pode ser usado independente da técnica de programação adotada. Terceiro, nós também investigamos meios para aliviar o esforço de manutenção quando

mudanças relacionadas ao código de composição precisam ser realizadas. Nesta investigação, nós avaliamos se modelos enriquecidos com a especificação das propriedades de composição ajudam os desenvolvedores a melhorar a estabilidade do programa em suas tarefas de manutenção.

Palavras-chave

Propriedades de Composição. Mecanismos de Composição. Estabilidade de Programas. Métricas de Software.

Table of Contents

1	Introduction	17
1.1	Problem Statement	19
1.1.1	The Problem of Characterizing Composition Properties	21
1.1.2	The Problem of Assessing the Impact of Composition Properties on Program Stability	22
1.1.3	The Problem of Alleviating the Maintainability Effort	23
1.2	Limitations of Related Work	24
1.2.1	Program Stability vs. Composition Properties	24
1.2.2	Composition Design	25
1.3	Goals and Research Questions	26
1.4	Contributions	28
1.5	Links between Research Questions, Papers and Chapters	30
1.6	Thesis Outline	30
2	Background and Related Work	34
2.1	Composition Mechanisms	35
2.1.1	Modularity with AOP Mechanisms	36
2.1.2	Modularity with FOP Mechanisms	37
2.1.3	Modularity with Other Advanced Programming Techniques	39
2.2	Program Stability and Modularity Metrics	41
2.2.1	Program Stability	41
2.2.2	Modularity Metrics	42
2.3	Empirical Studies on Stability vs. Advanced Programming Techniques	44
2.4	Composition Design	45
2.5	Summary	46
3	The Role of Modularity in Program Stability	48
3.1	Evaluation Methodology	50
3.1.1	Target Systems	50
3.1.2	Research Aims	52
3.1.3	Procedures	52
3.1.4	Modularity and Stability Metrics	53
3.2	Programming Techniques: Analysing Evolving Systems	55
3.2.1	Modularity Analysis	55
3.2.2	Stability Analysis	58
3.2.3	Discussion: Are Conventional Metrics Indicators of Program Stability?	62
3.3	Related Work	65
3.4	Threats to Validity	65
3.5	Summary	66
4	Composition Measurement Framework	68
4.1	Measurement Framework Requirements	69
4.2	Terminology and Composition Properties	70
4.2.1	Terminology	70

4.2.2	Composition Properties	73
4.3	The Measurement Suite	75
4.4	The Composition Properties Measurement Tool	80
4.4.1	CoMMes Architecture	80
4.4.2	User Interface	81
4.5	Application of the Measurement Framework	82
4.5.1	Goal and Hypotheses	83
4.5.2	Procedures	83
4.6	Framework Evaluation	85
4.6.1	Composition Properties vs. Stability	85
4.6.2	Coupling vs. Composition Measures	89
4.6.3	Other Framework Applications	92
4.7	Related Work	93
4.8	Known Limitations of the Framework	94
4.9	Threats to Validity	94
4.10	Summary	95
5	Design Models with Composition Properties	96
5.1	Motivating Example	97
5.2	UML+ Notation	98
5.3	Experimental Design	100
5.3.1	Research Questions and Hypotheses	101
5.3.2	Object	102
5.3.3	Task Design	103
5.3.4	Participants	105
5.3.5	Procedures	106
5.3.6	Variable and Analysis	108
5.4	Discussion	109
5.4.1	Statistical Test	109
5.4.2	Qualitative Discussion	112
5.4.3	Solution Quality meets Stability	114
5.5	Related Work	115
5.5.1	The Impact of UML Models	116
5.5.2	Analysis of Advanced Mechanisms	116
5.6	Threats to Validity	117
5.7	Summary	119
6	Final Considerations	120
6.1	Revisiting the Thesis Contributions	121
6.2	Future Work	122
	References	124
A	AppendixA - Target Applications	137
A.1	MobileMedia	137
A.2	iBatis	138
A.3	GameUP	138
B	AppendixB - Questionnaire of the Experiment	140

C	AppendixC - Experiment Details	143
C.1	Description of Tasks	143
C.2	Feedback Questionnaire	165

List of Figures

1.1	Composition Example using AspectJ notation	20
1.2	The relationships among RQs, Chapters and Main Publications	31
1.3	Thesis Overview	31
2.1	AOP in AspectJ	37
2.2	Virtual Classes	38
3.1	Modularity for iBatis	56
3.2	Modularity for MobileMedia	56
3.3	Modularity for Games	57
3.4	CaesarJ example of MobileMedia	60
3.5	AspectJ example	60
3.6	Modularization of features with virtual classes	60
3.7	Slice of code partially modified in CaesarJ	61
3.8	Example of modifications using Wrappers	62
3.9	Modularity vs Stability for CaesarJ	63
3.10	Modularity vs Stability for OO	63
3.11	DIT vs Stability for CaesarJ	64
4.1	Composition Code Measurement: Basic Terminology	70
4.2	Program in CaesarJ	71
4.3	Measurement Framework Overview	76
4.4	CoMMes Architecture	80
4.5	COM File Template	82
4.6	CoMMes output	83
4.7	Slice of iBatis Code (AspectJ)	87
4.8	Stability vs Gol for MobileMedia	87
4.9	Stability vs. CoV for MobileMedia	88
4.10	Composition Evolution	90
4.11	Coupling of the Composition Code (C1 and C5)	91
4.12	Stability of MobileMedia per modules	91
5.1	Design Model in UML+	99
5.2	Experiment Design	101
5.3	Average Expertise of the Participant Groups	106
5.4	The Experimental Overview	108
5.5	Average Percentage of Change Quality vs. Number of Answers Provided	112
5.6	Percentage of Changes realized by UML+ Group	115
C.1	UML Design for Task 1 - Class Diagram	145
C.2	UML Design for Task 1 - Sequence Diagram	146
C.3	UML+ Design for Task 1 - Class Diagram	147
C.4	UML+ Design for Task 1 - Sequence Diagram	148
C.5	UML Design for Task 2 - Class Diagram	151
C.6	UML Design for Task 2 - Sequence Diagram	152

C.7 UML+ Design for Task 2 - Class Diagram	153
C.8 UML+ Design for Task 2 - Sequence Diagram	154
C.9 UML Design for Task 3 - Class Diagram	156
C.10 UML Design for Task 3 - Sequence Diagram	157
C.11 UML+ Design for Task 3 - Class Diagram	158
C.12 UML+ Design for Task 3 - Sequence Diagram	159
C.13 UML Design for Task 4 - Class Diagram	161
C.14 UML Design for Task 4 - Sequence Diagram	162
C.15 UML+ Design for Task 4 - Class Diagram	163
C.16 UML+ Design for Task 4 - Sequence Diagram	164