

**Camila Patrícia Bazílio Nunes**

**History-Sensitive Recovery of Features in Code  
of Evolving Program Families**

**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. Carlos José Pereira de Lucena  
Co-advisor: Prof. Alessandro Fabricio Garcia

Rio de Janeiro  
October, 2012

**Camila Patrícia Bazílio Nunes**

## **History-Sensitive Recovery of Features in Code of Evolving Program Families**

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. Carlos José Pereira de Lucena**

Advisor

Departamento de Informática — PUC-Rio

**Prof. Alessandro Fabricio Garcia**

Co-advisor

Departamento de Informática — PUC-Rio

**Prof. Vander Alves**

UnB

**Prof. Uirá Kulesza**

UFRN

**Prof. Renato Fontoura de Gusmão Cerqueira**

Departamento de Informática - PUC-Rio

**Prof. Karin Breitman**

Departamento de Informática - PUC-Rio

**Prof. José Eugenio Leal**

Coordinator of the Centro Técnico Científico da PUC-Rio

Rio de Janeiro, October 19, 2012

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.

### **Camila Patrícia Bazílio Nunes**

She completed her undergraduate studies in Computer Science at the Federal University of Alagoas (UFAL) in 2006. She received her Master degree in Informatics from the Pontifical Catholic University of Rio de Janeiro (PUC-Rio) in 2009.

#### Bibliographic data

Nunes, Camila

History-Sensitive Recovery of Features in Code of Evolving Program Families / Camila Patrícia Bazílio Nunes; advisor: Carlos José Pereira de Lucena; co-advisor: Alessandro Fabricio Garcia. — Rio de Janeiro : PUC-Rio, Departamento de Informática, 2012.

155 f: il ; 30 cm

1. Tese (Doutorado em Informática) - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2012.

Inclui Bibliografia.

1. Informática – Tese. 2. Famílias de Programas Evolutivas. 3. Mapeamento de Características. 4. Heurísticas. 5. História Multi-Dimensional. I. Lucena, Carlos. II. Garcia, Alessandro. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

## Acknowledgments

First and foremost, I would like to deeply thank my supervisors Carlos Lucena and Alessandro Garcia for guiding me during the whole execution of the work. I am very grateful to Professor Carlos Lucena, who helped in the very beginning of my research career. In particular, my deepest thanks to Alessandro Garcia who always encouraged me and kept me on the right way.

I am also thankful to the members of my examining committee who have generously contributed their time and expertise: Vander Alves, Uirá Kulesza, Renato Cerqueira, Karin Breitman. Thanks also to all my colleagues and professors from the Computer Science Department at PUC-Rio. Many thanks to the Opus research group and the Software Engineering Laboratory (LES) at PUC-Rio. During my PhD, I also had the pleasure of working with other great researchers from different universities in the last four years. They have contributed a lot to this work by providing valuable suggestions that greatly improved the quality of my thesis.

I would like also to thank my family, my parents, Julio Nunes and Girleide Nunes, and my brothers, Sady and Leonardo. They always believed in me and encouraged me to go on this journey.

My doctoral studies were financially supported by CNPq and CAPES.

## Abstract

Nunes, Camila; Lucena, Carlos; Garcia, Alessandro. **History-Sensitive Recovery of Features in Code of Evolving Program Families**. Rio de Janeiro, 2012. 155p. DSc Thesis — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A program family might degenerate due to unplanned changes in its implementation, thus hindering the maintenance of family members. This degeneration is often induced by feature code of the program family that is changed individually in each member without considering other family members. In extreme cases, the program family code is fully or partially replicated and individually changed across several evolving members. Hence, as a family evolves over time, it might no longer be possible to identify and classify the implementation elements realizing common and variable features. One of the imminent activities to address these problems is the history-sensitive recovery of program family's features. This recovery process encompasses the historical analysis of each family member in order to identify and classify the implementation elements (i.e. methods, attributes) according to their variability nature. Existing work fails to analyse the evolution of the family members with the goal of recovering features' implementation elements. Additionally, existing techniques for feature analysis are not effective as they only take into consideration the history of a single member product. In summary, the contributions of this thesis are threefold: (i) a catalogue of mapping mismatches to guide software engineers in promoting the correctness and completeness of their feature mappings. This catalogue is useful to ensure a better effectiveness of the recovery process during the mapping analysis; (ii) a suite of five heuristics for the automatic expansion of feature mappings throughout the program family history. Those heuristics rely on both the multi-dimensional historical analysis of program families and the catalogue of mapping mismatches; and (iii) a suite of history-sensitive heuristics for classifying the implementation elements realizing each family feature according to their variability degree.

## Keywords

Evolving Program Families. Feature Mapping. Heuristics. Multi-dimensional history.

## Resumo

Nunes, Camila; Lucena, Carlos; Garcia, Alessandro. **Recuperação Sensível a História de Características no Código de Famílias de Programas Evolutivas**. Rio de Janeiro, 2012. 155p.

Tese de Doutorado — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Uma família de programas pode degenerar devido a mudanças não planejadas e, conseqüentemente, tendem a prejudicar a manutenção dos membros da família. Esta degeneração é frequentemente causada pelo código de uma característica (“feature”) da família que é modificada individualmente em cada membro sem considerar outros membros da família. Em casos extremos, o código da família é completamente ou parcialmente replicado e individualmente modificado por diversos membros em evolução. Assim, à medida que uma família evolui, pode não ser mais possível identificar e classificar os elementos de código implementando as características comuns e variáveis. Uma das atividades iminentes para resolver esses problemas é a recuperação sensível à história de características da família. Este processo de recuperação inclui a análise histórica de cada membro da família a fim de identificar e classificar os elementos de implementação (e.g. métodos, atributos) de acordo com a sua natureza de variabilidade. Os trabalhos existentes falham em analisar a evolução dos membros de uma família com o objetivo de recuperar os elementos de implementação das características. Além disso, as técnicas existentes para a análise de características não são efetivas, pois elas apenas levam em consideração a história de um único membro por vez. Em resumo, as contribuições desta tese são divididas em três partes: (i) um catálogo de incompatibilidades de mapeamento para guiar engenheiros de software na correção e completude de seus mapeamentos de características. Este catálogo é útil para garantir uma melhor eficácia do processo de recuperação durante a análise dos mapeamentos; (ii) um conjunto de cinco heurísticas para a expansão automática de mapeamentos de características ao longo do histórico da família de programas. Essas heurísticas são baseadas na análise histórica multi-dimensional da família e no catálogo de incompatibilidades de mapeamentos; e (iii) um conjunto de heurísticas sensíveis a história para classificar os elementos de implementação de cada característica da família de acordo com seu grau de variabilidade.

## Palavras-chave

Famílias de Programas Evolutivas. Mapeamento de Características. Heurísticas. História Multi-Dimensional.

# Table of Contents

1	Introduction	11
1.1	Problem Statement	13
1.1.1	Degeneration of Program Families	13
1.1.2	Examples of Degeneration Symptoms	15
1.1.3	How to Recover Feature Code in Degenerate Families?	17
1.2	The State of the Art on Design Recovery and Feature Mapping	18
1.3	Aims and Research Questions	20
1.4	Thesis Contributions	21
1.4.1	A Catalogue of Recurring Feature Mapping Mismatches (RQ1)	22
1.4.2	A Suite of Mapping Expansion Heuristics (RQ2)	22
1.4.3	Recovery Heuristics for the Classification of Feature Elements (RQ3)	23
1.4.4	Empirical Evaluation (RQ4)	23
1.5	Outline of the Thesis Structure	25
2	Background and Related Work	27
2.1	Terminology	28
2.2	Feature Identification Support	29
2.2.1	Feature Mapping Activity	30
2.2.2	Static Techniques	32
2.2.3	Dynamic Techniques	33
2.2.4	Hybrid Techniques	33
2.2.5	The Differences of the Feature Mapping Techniques	34
2.3	Reengineering Methodologies and Techniques	35
2.4	Source Code Refactoring and History Analysis	37
2.5	Clone Detection Techniques	40
2.6	Summary	41
3	Mismatches in Feature Mappings	42
3.1	Identifying Mapping Mismatches	44
3.1.1	Target Program Family	44
3.1.2	Study Procedures	45
3.2	Catalogue of Mismatches in Feature Mappings	46
3.2.1	Feature Characteristics	46
3.2.2	Module Characteristics	51
3.3	Correlating the Mapping Mismatches	53
3.4	Experimental Evaluation	55
3.4.1	Selected Software Systems and Features	55
3.4.2	Experimental Procedures	56
3.4.3	Quantifying the Mapping Mismatches	57
3.4.4	Threats to Validity	61
3.5	Limitations of Related Work	62
3.5.1	Feature Mapping Studies	62
3.5.2	Mapping Mismatches and Existing Techniques	63
3.6	Summary	64

4	Mapping Expansion Heuristics	<b>66</b>
4.1	Motivating Example	67
4.2	Existing Limitations on Feature Mapping Expansion	70
4.2.1	Feature Mapping Techniques	70
4.2.2	Source Code History	70
4.3	A Heuristic Method for Expanding Feature Mappings	71
4.3.1	Multi-dimensional History Analysis	72
4.3.2	Comparison Strategy and Mapping Expansion Process	74
4.4	Mapping Expansion Heuristics	75
4.4.1	Execution Order of the Heuristics	76
4.4.2	Detecting Omitted Feature Partitions	76
4.4.3	Detecting Code Clone Mismatches	81
4.4.4	Detecting Interfaces and Super-Classes	83
4.4.5	Detecting Communicative Feature Mismatches	86
4.4.6	Detecting Omitted Attributes	88
4.5	MapHist: A Heuristic-based Tool for Expanding Feature Mappings	91
4.5.1	The MapHist Architecture	91
4.5.2	Representation of the Implementation Elements and Use of Existing Tools	93
4.6	Evaluation	94
4.6.1	Target Program Families	94
4.6.2	Study Design	96
4.6.3	Data Analysis	99
4.6.4	Discussions	104
4.6.5	Limitations	105
4.7	Threats to Validity	106
4.8	Integration of the Mapping Heuristics with a Visualization Tool	107
4.9	Summary	110
5	Recovery Heuristics of Feature Elements	<b>112</b>
5.1	Recovery Methodology	113
5.2	Forward Recovery Heuristics	114
5.2.1	Recovering Elements in Common	114
5.2.2	Recovering Variable Elements	120
5.3	Algorithm Solution and Implementation	123
5.4	Assessment Methodology	125
5.5	Discussion	126
5.6	Usefulness of the Proposed Categories	130
5.7	Threats to Validity	132
5.8	Summary	133
6	Final Remarks and Future Work	<b>135</b>
6.1	Revisiting the Thesis Contributions	136
6.2	Future Work	138
	References	<b>141</b>



## List of Figures

1.1	Evolution Strategy using SVN.	14
1.2	Modification of Framework Code Elements with Variable Code.	16
1.3	The Implementation of Member-specific Code is Intermingled with Core Code.	17
3.1	Multi-Partition Feature.	47
3.2	Overly Communicative Features.	48
3.3	Mapping Mismatches Relationships.	54
3.4	Mapping Mismatches in both Health Watcher and MobileMedia systems.	58
4.1	Piece of Code of the Feature Scenario in Applications I and II.	68
4.2	Multi-Dimensional History of a Program Family Evolution.	73
4.3	Methodology of the Mapping Expansion Heuristics.	73
4.4	Example of Interaction Similarity.	77
4.5	Abstract Representation of the Original Heuristic and DFP.	80
4.6	Abstract Representation of DCC.	83
4.7	Abstract Representation of DIS.	85
4.8	Abstract Representation of DCF.	88
4.9	Abstract Representation of DOA.	91
4.10	The MapHist Architecture.	92
4.11	Number of Elements throughout the History in OC program family.	103
4.12	Workflow of the Mapping Expansion Heuristics.	104
4.13	Proactive and Interactive Visualization Strategy.	108
4.14	Evolution of the feature Transaction in the Treemap view.	110
5.1	Methodology of the Recovery Heuristics.	113
5.2	Illustrative Example of Full Vertical and Horizontal Similarity.	116
5.3	Illustrative Example of Full Vertical Similarity.	116
5.4	Illustrative Example of Full Horizontal Similarity.	117
5.5	Illustrative Example of Partial Vertical and Horizontal Similarity.	118
5.6	Feature Mappings of a Program Family.	119
5.7	Illustrative Example of Horizontal Variability.	121
5.8	Illustrative Example of Vertical Variability.	121
5.9	Java Project of the Recovered Program Family.	123
5.10	The Forward Recovery Implementation.	124
5.11	Precision (P) and Recall (R) of the Full (F), Partial (Pt), and Variable (V) Categories.	127

## List of Tables

1.1	Publications related to this Thesis.	24
1.2	Indirect Publications.	24
2.1	Differences among the Feature Mapping Techniques.	35
3.1	Features analysed in OC Program Family.	44
3.2	Features analysed in Health Watcher and MobileMedia systems.	56
4.1	Analysed features in OC.	96
4.2	Analysed features in RAWeb.	96
4.3	Precision (P) and Recall (R) Results for each Version of the OC framework (OC family).	100
4.4	Precision (P) and Recall (R) Results for each Version of Application I (OC family).	101
4.5	Precision (P) and Recall (R) Results for each Version of Application II (OC family).	101
4.6	Precision (P) and Recall (R) Results for each Version of Application III (OC family).	101
4.7	Precision (P) and Recall (R) Results for each Version of Application I (RAWeb family).	102
4.8	Precision (P) and Recall (R) Results for each Version of Application II (RAWeb family).	102
4.9	Precision (P) and Recall (R) Results for each Version of Application III (RAWeb family).	102