



**Everton Tavares Guimarães**

**A Blueprint-Based Approach for  
Prioritizing and Ranking Critical  
Code Anomalies**

**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  
September 2014



**Everton Tavares Guimarães**

**A Blueprint-Based Approach for  
Prioritizing and Ranking Critical  
Code Anomalies**

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. Julio Cesar Sampaio do Prado Leite**

Departamento de Informática – PUC-Rio

**Prof. Alberto Raposo Tavares**

Departamento de Informática – PUC-Rio

**Prof. Thais Vasconcelos Batista**

UFRN

**Prof. Eduardo Magno Lages Figueiredo**

UFMG

**Prof. José Eugenio Leal**

Coordinator of the Centro Técnico Científico da PUC-Rio

Rio de Janeiro, September 12th, 2014

All rights reserved.

## Everton Tavares Guimarães

He completed his undergraduate studies in the Federal Institute of Science, Education and Technology of Rio Grande do Norte in 2009. He received his Master degree in Computer Science from the Federal University of Rio Grande do Norte (UFRN) in 2010. His research topics include software evolution, software product lines, software modularity, software metrics, aspect-oriented software development, model-driven development, code anomalies and empirical studies.

### Bibliographic data

Guimarães, Everton Tavares

A Blueprint-Based Approach for Prioritizing and Ranking  
Critical Code Anomalies / Everton Tavares Guimarães; advisor:  
Alessandro Fabrício Garcia – 2014.

142 f. : il. ; 30 cm

Tese (Doutorado em Informática)–Pontifícia Universidade  
Católica do Rio de Janeiro, Rio de Janeiro, 2014.

Inclui referências bibliográficas

1. Informática – Teses. 2. Anomalias de Código. 3. Projeto  
Arquitetural. 4. Blueprints. 5. Métricas de Software. 5. Estudos  
Empíricos. 5. Heurísticas Sensíveis a Arquitetura. I. Garcia,  
Alessandro Fabrício. II. Pontifícia Universidade Católica do Rio de  
Janeiro. Departamento de Matemática. IV. Título.

CDD: 004

To all my friends, family and fiancée.  
To all dear colleagues at the OPUS Research Group.  
To my advisor, who guided me through this uneasy research  
for the last four years. Thank you for your patience,  
kindness and wisdom.

## Acknowledgments

First of all, I would like to thank God for all the blessings in my life and for giving me strength and persistence to overcome all the challenges and barriers I have found in the last four years.

I would like to express my sincere gratitude to Prof. Alessandro Garcia, who agreed to be my supervisor and helped me in preparing this thesis despite his busy schedule. Thank you for your availability, patience, and for all the knowledge and experience you have shared with me during this journey. His professionalism, dedication and energy for work are exemplary. Above all, thanks for the friendly relationship built over these four years.

I would like to thank my parents, Alvaro and Marilsa, for all the love and support. They have not supported me in this journey, but also for all the many other challenges in my life. They have been especially important and played a critical role in the completion of this thesis. I have no words to describe how blessed I am to be your son.

I would like to express my gratitude to my fiancée Annanda for being by my side either in the good or bad moments. Her dedication, discipline and patience are inspiring. Your love, affection and understanding were the driving force to keep me I reached this goal (specially during the last year).

Thanks to all my friends at PUC-Rio, and many other friends I have met in Rio. I would like to thank for your friendship. It is great pleasure for me to be in your presence and enjoy. Meet them in his spare time are always a boost to continue working. I wish I could name all my friends and loved ones who have been there for me during this journey graduate school, but there are simply too many to list. However, I would like to thank Michele and José for all the support, friendship, and advices. Thanks for and for being my friends unconditionally.

I would like to thank all friends of courses taken in the master for the fellowship, encouragement in difficult times and relaxed moments. Without your company, my journey would have been much more difficult.

I would like to thank all the friends and colleagues of LES and the members of the OPUS research group. Your comments and discussions are always enriching. In particular, thanks to Soeli, Francisco, Bruno, Eiji, and Isela who has also participated with me in many research projects, and therefore, have more closely contributed to my doctoral research.

I would like to thank all the teachers of the Department of Informatics, PUC-Rio for the contribution in my education. I also thank all the staff of the department for their services.

## Abstract

Guimarães, Everton Tavares; Garcia, Alessandro Fabricio, Alessandro Fabrício (Advisor). **A Blueprint-Based Approach for Prioritizing and Ranking Critical Code Anomalies**, 2014. 142p. DSc Thesis - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Software systems are often evolving due to many changing requirements. As the software evolves, it grows in size and complexity, and consequently, its architecture design tends to degrade. Architecture degradation symptoms are often a direct consequence of the progressive insertion of code anomalies in the software implementation. A code anomaly is a recurring implementation structure that possibly indicates deeper architectural design problems. Code anomaly is considered critical when it is related with a structural problem in the software architecture. Its criticality stems from its negative influence on a wide range of non-functional requirements. For instance, the presence of critical code anomalies hinders software maintainability, i.e. these critical anomalies require wide refactoring in order to remove an architectural problem. Symptoms of architecture degradation have often to be observed in the source code due to the lack of an explicit, formal representation of the software architecture in a project. Many approaches are proposed for detecting code anomalies in software systems, but none of them efficiently support the prioritization and ranking of critical code anomalies according to their architecture impact. Our work investigates how the prioritization and ranking of such critical code anomalies could be improved by using blueprints. Architecture blueprints are usually provided by software architects since the early stages of the system development. Blueprints are informal design models usually defined to capture and communicate key architectural design decisions. Even though blueprints are often incomplete and inconsistent with respect to the underlying implementation, we aim to study if their use can contribute to improve the processes of prioritizing and ranking critical code anomalies. Aiming to address these research goals, a set of empirical studies has been performed. We also proposed and evaluated a set of

heuristics to support developers when prioritizing and ranking code anomalies in 3 software systems. The results showed an average accuracy higher than 60% when prioritizing and ranking code anomalies associated with architectural problems in these systems.

## **Keywords**

Code Anomalies; Architecture Design; Blueprints; Software Metrics;  
Empirical Studies; Architecture Sensitive Heuristics;



## Resumo

Guimarães, Everton Tavares; Garcia, Alessandro Fabricio. **Uma Abordagem Baseada em Blueprints para Priorização e Classificação de Anomalias de Código Críticas**. Rio de Janeiro, 2014. 142p. Tese de Doutorado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Sistemas de software estão evoluindo frequentemente devido a diversas requisições de mudanças. A medida que o software evolui, seu tamanho e complexidade aumentam, e conseqüentemente, sua arquitetura tende a se degradar. Sintomas de degradação arquitetural são por muitas vezes uma consequência direta da inserção progressiva de anomalias de código. Uma anomalia de código é uma estrutura da implementação recorrente que possivelmente indica problemas mais severos no projeto arquitetural. Anomalia de código é considerada crítica quando ela está relacionada problemas estruturais na arquitetura do software. Sua criticidade origina-se da sua influência negativa em uma ampla gama de requisitos não-funcionais. Por exemplo, a presença e anomalias e código críticas dificulta a manutenibilidade e software., ex. uma grande refatoração pode ser necessária para remover um problema arquitetural. Diversas abordagens tem sido propostas para a detecção de anomalias em sistemas de software, mas nenhuma delas suporta eficientemente a priorização e classificação de anomalias de código críticas de acordo com seu impacto na arquitetura. O presente trabalho investiga como a priorização e classificação dessas anomalias críticas de código pode se melhorado com o uso de *blueprints* arquiteturais. *Blueprints* arquiteturais são providos pelo arquiteto de software desde estágios iniciais de desenvolvimento do sistema. *Blueprints* são modelos de projeto informais normalmente definidos para capturar e comunicar as principais decisões de projeto arquitetural. Embora *blueprints* normalmente sejam incompletos e inconsistentes com respeito a implementação do sistema, eles podem contribuir para o processo de priorização e classificação de anomalias de código críticas. Com o intuito de alcançar nossos objetivos de pesquisa, um conjunto de estudos empíricos foram realizados. O trabalho também propõe e avalia um conjunto de heurísticas para auxiliar

desenvolvedores na priorização e classificação de anomalias de código em 3 sistemas de software. Os resultados mostraram uma acurácia média de mais de 60% na priorização e classificação de anomalias de código associadas com problemas arquiteturais nesses sistemas.

## **Palavras-Chave**

Anomalias de Código; Projeto Arquitetural; Blueprints; Métricas;  
Estudos Empíricos; Heurísticas Sensíveis a Arquitetura;

# Table of Contents

1 Introduction	16
1.1. Motivation	17
1.2. Problem Statement	21
1.3. Limitations of Related Work	23
1.4. Proposed Solution	24
1.5. Research Questions	28
1.6. Outline of the Thesis Structure	30
2 Background and Related Work	33
2.1. Basic Terminology	34
2.1.1. Software Architecture	34
2.1.2. Architecture Degradation Symptoms	35
2.2. Code Anomalies as Architecture Degradation Symptoms	38
2.2.1. Refactoring Process and Removal of Code Anomalies	39
2.2.2. Co-occurrence of Critical Code Anomalies	40
2.2.3. Classifying Occurrences of Code Anomalies	42
2.3. Tool Support for Code Anomaly Detection	43
2.3.1. Prioritization and Ranking of Code Anomalies	44
2.4. Prioritization of Code Anomalies Supported by Blueprints	46
2.4.1. Level of Abstraction	48
2.4.2. Completeness	49
2.4.3. Inconsistencies	50
2.4.4. Mapping Architecture Blueprints to Source Code	51
2.5. Summary	53
3 On the Relation of Blueprints and Code Anomalies: A Study of Evolving Software Systems	55
3.1. Aspect-Oriented Software Development	56
3.2. Aspect-Oriented Modeling	59
3.3. Study Settings	61
3.3.1. Target Systems	62

3.3.2. Quantifying Modularity Properties and Inconsistencies	63
3.3.3. Study Phases and Assessment Procedures	64
3.4. Hypotheses Testing and Initial Research Findings	66
3.5. Code Anomalies and Inconsistencies in the Descriptive Architecture	68
3.6. Summary	73
4 Prioritizing and Ranking Code Anomalies with Blueprints	75
4.1. Code Anomalies Studied	76
4.2. Experimental Evaluation	77
4.2.1. Experimental Steps	79
4.2.2. Ground Truth of Code Anomalies	81
4.3. Hypotheses Testing and Data Analysis	81
4.3.1. Impact of Architecture Blueprints On Precision and Recall	83
4.3.2. Analyzing the Time Spent on the Prioritization Process	84
4.4. Further Discussions	85
4.4.1. Usefulness of Architecture Blueprints	86
4.4.2. Participants' Technical Knowledge	87
4.4.3. Mapping Architecture Blueprints to Source-Code	88
4.4.4. Critical Code Anomalies and False Positives	89
4.4.5. Architecture Blueprints and False Negatives	92
4.5. Summary	93
5 Architecture Sensitive Heuristics for Prioritizing and Ranking Critical Code Anomalies	96
5.1. Criteria Selection and Relation to Architectural Problems	98
5.2. Study Settings	99
5.3. Heuristics for Prioritizing and Ranking Critical Code Anomalies	102
5.3.1. Inter-Component Heuristics	102
5.3.1.1. Heuristic Based External Attractor Component	103
5.3.1.2. Heuristic Based on External Addictor Component	105
5.3.2. Concern-Based Heuristics	106
5.3.2.1. Heuristic for Concern Overload	107
5.3.2.2. Heuristic for Misplaced Concern	109
5.4. Research Findings on Prioritizing and Ranking Critical Code Anomalies	110
5.4.1. Procedures for Data Collection and Evaluation Method	110

5.4.2. Target Applications	113
5.4.3. Inter-Component Heuristics	114
5.4.4. Concern-Based Heuristics	116
5.4.5. Accuracy of the Architecture Sensitive Heuristics	119
5.5. Discussions	120
5.5.1. Identifying False Positives and False Negatives.	121
5.5.2. Comparing Ranking Provided by Different Heuristics	122
5.6. Summary	124
6 Final Remarks	127
6.1. General Threats to Validity	127
6.2. Revisiting the Thesis Contributions	128
6.3. Future Works	132
References	135

## List of Figures

Figure 1 - Code Fragments with Non-Critical and Critical Code Anomalies.....	19
Figure 2 - Architecture Blueprint of Mobile Media SPL.....	48
Figure 3 - Mapping Component to Source-Code.....	52
Figure 4 – AOM notation for architecture blueprints .....	59
Figure 5 – Information Suggested by Participants.....	59
Figure 6 – Scenario for External Attractor Component.....	104
Figure 7 – Scenario for External Addictor Component.....	106
Figure 8 - Scenario for Misplaced Concern.....	108

## List of Tables

Table 1 - Architectural problems considered in this work.....	36
Table 2 - Correlation between Inconsistencies and AO Modularity Properties .....	67
Table 3 – Anomalies in AO Target Application .....	71
Table 4 – Code Anomalies in Health Watcher and Mobile Media.....	71
Table 5 – Correlation between Code Anomalies and Inconsistencies.....	72
Table 6 – Study hypotheses definition.....	78
Table 7 – Code anomalies reference list for Mobile Media.....	81
Table 8 - Definition of Precision and Recall .....	82
Table 9 – Descriptive statistic for Precision and Recall .....	83
Table 10 – Descriptive statistic for Time Spent on the prioritization process.....	85
Table 11 - Analyzing consistency and Completeness of architecture blueprint.....	89
Table 12 - Characteristics of False Positives .....	91
Table 13 - Software Metrics and False Positives.....	92
Table 14 - False Negatives by Participants' Group.....	93
Table 15 - Study Definition using GQM format.....	101
Table 16 - Study Hypothesis for Evaluating the Heuristics.....	101
Table 17 - Additional Architecture Sensitive Metrics .....	112
Table 18 - Characteristics of Target Applications .....	113
Table 19 - Results for Inter-Component Heuristics.....	115
Table 21 – Architectural Concerns for Health Watcher and Mobile Media.....	119
Table 21 – Results for Concern Based Heuristics .....	119
Table 22 - False positives and negatives achieved by the prioritization heuristics ...	121
Table 23 – Relevance based on the system history evolution .....	123