Congresso Brasileiro de Software: Teoria e Prática
28 de setembro a 03 de outubro de 2014 - Maceió/AL

ICSE 2017 PhD and Young Researchers Warm Up Symposium

ICSE 2017 SYMPOSIUM

Anais
Anais

ICSE 2017 SYMPOSIUM
ICSE 2017 PhD and Young Researchers Warm Up Symposium

COORDENADORES DO ICSE 2017 SYMPOSIUM
Rohit Gheyi - Universidade Federal de Campina Grande (UFCG)
Marcio Ribeiro - Universidade Federal de Alagoas (UFAL)
Alessandro Garcia - Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
Sebastian Uchitel - Imperial College London and Universidad de Buenos Aires

COORDENAÇÃO DO CBSOFT 2014
Baldoino Fonseca - Universidade Federal de Alagoas (UFAL)
Leandro Dias da Silva - Universidade Federal de Alagoas (UFAL)
Márcio Ribeiro - Universidade Federal de Alagoas (UFAL)

REALIZAÇÃO
Universidade Federal de Alagoas (UFAL)
Instituto de Computação (IC/UFAL)

PROMOÇÃO
Sociedade Brasileira de Computação (SBC)

PATROCÍNIO
CAPES, CNPq, INES, Google

APOIO
Instituto Federal de Alagoas, Aloo Telecom, Springer, Secretaria de Estado do Turismo AL, Maceió Convention & Visitors Bureau, Centro Universitário CESMAC e Mix Cópia
ICSE 2017 SYMPOSIUM

ICSE 2017 PhD and Young Researchers Warm Up Symposium

PROGRAM CHAIRS
Rohit Gheyi - Universidade Federal de Campina Grande (UFCG)
Marcio Ribeiro - Universidade Federal de Alagoas (UFAL)
Alessandro Garcia - Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
Sebastian Uchitel - Imperial College London and Universidad de Buenos Aires

CBSOFT 2014 GENERAL CHAIRS
Baldoino Fonseca - Universidade Federal de Alagoas (UFAL)
Leandro Dias da Silva - Universidade Federal de Alagoas (UFAL)
Márcio Ribeiro - Universidade Federal de Alagoas (UFAL)

ORGANIZATION
Universidade Federal de Alagoas (UFAL)
Instituto de Computação (IC/UFAL)

PROMOTION
Sociedade Brasileira de Computação (SBC)

SPONSORS
CAPES, CNPq, INES, Google

SUPPORT
Instituto Federal de Alagoas, Aloo Telecom, Springer, Secretaria de Estado do Turismo - AL, Maceió Convention & Visitors Bureau, Centro Universitário CESMAC and Mix Cópia
Autorizo a reprodução parcial ou total desta obra, para fins acadêmicos, desde que citada a fonte
Foreword

We are extending the deadline since a number of students do not notice that this is a semi-open event and they need to submit an abstract with 1,800 characters. This is a semi-open event and we intend to select approximately 20 participants to attend the symposium. It is easier for invited speakers to provide frank feedback, for students to expose their worries, problems and doubts, and so no, in a semi-open event.

In 2017 the International Conference on Software Engineering (ICSE) will be held in South America for the first time since it started in 1975. More specifically, ICSE will be held in Buenos Aires around May 2017. ICSE is the premier software engineering conference, providing a forum for researchers, practitioners and educators to present and discuss the most recent innovations, trends, experiences and concerns in the field of software engineering.

We believe that up to now South American presence at ICSE does not reflect the quality and potential of software engineering research that is ongoing in the region. We believe that ICSE 2017 in South America is a unique opportunity for changing this, not only bringing into the ICSE community South American researchers, practitioners and educators but also inspiring and boosting the potential of the budding South American software engineering community. Furthermore, we believe that ICSE 2017 in South America can also serve as a catalyst, fostering stronger integration between software engineering research communities of South American countries.

The ICSE 2017 PhD and Young Researchers Warm Up Symposium aims to bring senior year PhD students and young PhD researchers together to share their experience and to learn from established academic and researchers from the International Conference on Software Engineering community. Developing as a young scientist and achieving impact through publications in top scientific venues such as ICSE is a challenge that not only requires creativity, well articulated hypothesis and rigorous validation but also communication skills, both for written and oral presentations, understanding how communities work and how to get results disseminated.

The symposium focuses on helping new software engineering researchers feel more comfortable and confident in dealing with these many challenges. The symposium will feature advice and guidance from leading software engineering researchers based upon their personal experiences and insights into the contemporary community. Ample time for informal and small group interactions will allow the attendees to dig deeper into pertinent questions and concerns.

A goal of this symposium will be to provide a supportive yet questioning setting in which young researchers can present their work. Participants will be able to discuss their goals, methods, and results at an early stage in their research and receive useful guidance and feedback on various aspects of their research. The symposium will also
help participants to establish a research and social network with their local peers and also with prominent members of the ICSE community. This year we are fortunate to have four previous ICSE Program Committee chairs attending the symposium!

Supported by ACM SIGSOFT
Co-located with CBSoft 2014
October 3, Maceió - Brazil
Comitê organizador / Organizing Committee

COORDENAÇÃO GERAL
Baldoino Fonseca - Universidade Federal de Alagoas (UFAL)
Leandro Dias da Silva - Universidade Federal de Alagoas (UFAL)
Márcio Ribeiro - Universidade Federal de Alagoas (UFAL)

COMITÊ LOCAL
Adilson Santos - Centro Universitário Cesmac (CESMAC)
Elvys Soares - Instituto Federal de Alagoas (IFAL)
Francisco Dalton Barbosa Dias - Universidade Federal de Alagoas (UFAL)

COORDENADORES DO ICSE 2017 SYMPOSIUM
Rohit Gheyi - Universidade Federal de Campina Grande (UFCG)
Marcio Ribeiro - Universidade Federal de Alagoas (UFAL)
Alessandro Garcia - Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
Sebastian Uchitel - Imperial College London and Universidad de Buenos Aires
<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Recommendation System for Repairing Software Architecture Erosion</td>
<td>12</td>
</tr>
<tr>
<td>Ricardo Terra</td>
<td></td>
</tr>
<tr>
<td>Enhancing Conformance Checking for Contract-Based Programs</td>
<td>14</td>
</tr>
<tr>
<td>Alysson Milanez</td>
<td></td>
</tr>
<tr>
<td>Evolving C Program Families Safely</td>
<td>16</td>
</tr>
<tr>
<td>Flavio Medeiros</td>
<td></td>
</tr>
<tr>
<td>Characterizing Product Line Architecture Recovery</td>
<td>18</td>
</tr>
<tr>
<td>Crescencio Rodrigues Lima Neto</td>
<td></td>
</tr>
<tr>
<td>Classifying Metrics For Assessing Object-Oriented Software</td>
<td>19</td>
</tr>
<tr>
<td>Maintainability: A Family Of Metrics’ Catalogs</td>
<td></td>
</tr>
<tr>
<td>Juliana Saraiva</td>
<td></td>
</tr>
<tr>
<td>Privacy and Security Policies for Collaborative Crosscutting Tasks</td>
<td>21</td>
</tr>
<tr>
<td>Rodrigo Andrade</td>
<td></td>
</tr>
<tr>
<td>Characterizing Code Smell Detection</td>
<td>23</td>
</tr>
<tr>
<td>José A. M. Santos</td>
<td></td>
</tr>
<tr>
<td>Parallel Development of Tasks</td>
<td>24</td>
</tr>
<tr>
<td>Paola Accioly</td>
<td></td>
</tr>
<tr>
<td>A new approach to bolster Software Engineering Learning Curve</td>
<td>26</td>
</tr>
<tr>
<td>Troy Kohwalter</td>
<td></td>
</tr>
<tr>
<td>An Approach for Analyzing Architectures Using Architectural Properties in SysML</td>
<td>27</td>
</tr>
<tr>
<td>Daniel Cunha Da Silva</td>
<td></td>
</tr>
<tr>
<td>Developers Assignment for Merging Branches</td>
<td>28</td>
</tr>
<tr>
<td>Catarina Costa</td>
<td></td>
</tr>
<tr>
<td>Detecting Behavioral Changes in Refactorings of Concurrent Programs</td>
<td>29</td>
</tr>
<tr>
<td>Felipe Pontes</td>
<td></td>
</tr>
<tr>
<td>Sustainable Software Evolution for Ecosystem Architectures</td>
<td>31</td>
</tr>
<tr>
<td>Simone Da Silva Amorim</td>
<td></td>
</tr>
</tbody>
</table>
Continuous Software Development: Unifying development and production environments
Rodrigo A. Vilar

A contribution to the definition and automation of structural testing of autonomous vehicles
Vânia Neves

Safe Evolution of Software Product Lines
Leopoldo Teixeira

Cookbooks for Framework Instantiation: Semi-automated Construction Driven by Examples
Raquel Lafetá

Evaluating and Redesigning Software Applications to Meet Usability Standards and Improve User eXperience Early in the Development Process
Luis Jorge Enrique Rivero Cabrejos

An Approach for Knowledge Management to Facilitate Improvements in Software Organizations
Jacliane Rabelo

On the Reconstruction of High-Level Software Abstractions from Execution Traces
Liliane Do Nascimento Vale

Conquering Exceptions with a Recommender System
Eiji Adachi Barbosa

Identification and Management of Technical Debt using Software Visualization Resources
Thiago Mendes

Bug Reopening: Impact of Process and Design Variables
Rodrigo Souza

A Recommendation System for Product Configuration in Software Product Lines
Juliana Pereira

Certification-Based Development Methodology of Biomedical Signal Acquisition Systems
Alvaro Sobrinho

Using CBAR Concepts to Automate Test Oracles for TTS systems
Rafael Oliveira
On the Synthesis of Architecturally-relevant Code Anomalies
Willian Oizumi

Supporting newcomers overcoming barriers to contribute to Open Source Software projects
Igor Steinmacher

How can visualization help developers improve exception handling design?
Benjamin Jakobus

Do the refactoring activities improve software maintainability?
Diego Cedrim

Analyzing and Supporting Behavior Preservation in Software Refactorings
Leonardo Sousa

On Leveraging Interactive Detection of Code Anomalies
Danyllo Albuquerque

Understanding the Time Effectiveness of Pairs in Identifying Code Anomalies
Roberto Oliveira

Towards a Process Models to Conduct Software Engineering Experiments
Liliane Fonseca

Crowdsourcing: Software Industry transformation and disruption
Leticia Santos Machado

Using social, history and technical aspects from software development to explain the occurrence of co-changes
Igor Scaliante Wiese

On the evaluation of Distributed Collaborative Programming
Bernardo Estácio

Safe Monitoring of Physical Exercises
Elthon Oliveira

Exploring Architecture Blueprints for Prioritizing Critical Code Anomalies
Everton Guimaraes
Improving the usability through the creation of models in early stages of the software process
Natasha Valentim

An Approach for History-Sensitive Reliability Verification for Software Product Lines
Andre Luiz Peron Martins Lanna

Strategies for Prioritizing Test Cases Generated Through Model-Based Testing Approaches
Joao Felipe Silva Ouriques

Towards Effort Estimation in Software Testing Projects
Thiago Silva-De-Souza
A Recommendation System for Repairing Software Architecture Erosion

Ricardo Terra
Young Researcher at Federal University of Lavras (UFLA), Lavras, MG, Brazil, terra@dcc.ufla.br, WWW home page: http://www.dcc.ufla.br/~terra

Abstract

Architectural erosion is a recurrent problem faced by software architects. Although several architecture conformance techniques have been proposed to detect architectural violations (e.g., reflection models, dependency structure matrices, source code query languages, constraint languages, architecture description languages, and design tests), less research effort has been dedicated to the task of repairing such violations. As a consequence, developers usually perform the repairing task in ad hoc ways, without tool support at the architectural level. This fact may lead developers to spend a long time to discover how to repair the architectural violations and even to introduce new violations when repairing one.

In view of such circumstances, this thesis proposes an architectural repair recommendation system that provides fixing guidelines for developers and maintainers when tackling architectural erosion. We have formalized a catalog of repairing recommendations to repair violations raised by static architecture conformance checking approaches; we have elaborated a suitable module heuristic to determine the correct module for source code entities; we have designed a tool—called ArchFix—that triggers the proposed recommendations; and we have evaluated the application of our approach in two industrial-strength systems. For the first system—a 21 KLOC open-source strategic management system—our approach indicated correct repairing recommendations for 31 out of 41 violations detected as the result of an architecture conformance process. For the second system—a 728 KLOC customer care system used by a major telecommunication company—our approach triggered correct recommendations for 632 out of 787 violations, as asserted by the system’s architect. Moreover, the architects scored 80% of these recommendations as having moderate or major complexity.
Short Bio (Ph.D. Student from March 2009 to September 2013)

Ricardo Terra holds a Ph.D. in Computer Science at Federal University of Minas Gerais, Brazil (DCC/UFMG). He was under the supervision of Prof. Roberto Bigonha and Prof. Marco Tulio Valente. Also, he’s done a 1-year internship under supervision of Prof. Krzysztof Czarnecki at University of Waterloo, Canada.

Nowadays, he is a young researcher of the Department of Computer Science at Federal University of Lavras (DCC/UFLA). Software Architecture is his main research topic. More precisely, he’s been working on approaches to address the software architecture erosion problem.

Since his admission to the M.Sc. in 2007, Ricardo Terra has published 25 articles in conferences and journals (IEEE Software, Software: Practice and Experience, CSMR, WCRE, ICPC, MSR, etc.)¹, co-supervised two M.Sc. students, and received the following honors: (i) 3rd best Master’s Thesis from CLTM/CLEI, (ii) one of the 11 best Master’s Theses from CTD/SBC, (iii) one award for the best article, and (iv) two awards for second best tool.

Acknowledgement

I’d like to thank the committee members for their time.

¹ See the complete list of publications at: www.ricardoterra.com.br/publications
Enhancing Conformance Checking for Contract-Based Programs

Alysson Milanez

Department of Computing Systems
Federal University of Campina Grande, Brazil
alyssonfilgueira@copin.ufcg.edu.br

Abstract. Testing is commonly used to check conformance in contract-based programs, as verification by formal proofs is hard to scale and static analysis is, sometimes, limited for detecting general nonconformances. Traditional test cases, with manually-provided data, may be ineffective in detecting subtle nonconformances that arise only after several instantiations and modifications in objects under test. Random-generated tests with automatic test data generation, on the other hand, is a promising approach more substantial testing is demanded. In my Masters degree, I propose and evaluate an approach, JMLOK 2.0, for automatically detecting and categorizing nonconformances, in the context of Java Modeling Language (JML), aiming to help the programmer in the process of nonconformances correction. The detection is backed by Randomly-Generated Tests (RGT) approach. And the categorization is backed by heuristics-based approach. Two evaluations were performed. First, an evaluation of the detection approach and the manual categorization process: 84 nonconformances were detected in over 29 KLOC and 9 K lines of JML contracts; applying the manual classification system I got most detected nonconformances were classified as postcondition errors. Second, an evaluation of the automatic categorization approach was performed: a comparison between automatic and manual categorization, getting a matches value of 0.73 (considering the 84 nonconformances from first evaluation) indicating similarity between automatic and manual approaches. Currently, in my PhD I’m working on nonconformances categorization in the context of Spec# programs.

Short bio
I am Alysson Milanez, currently a PhD student in Computer Science at the Federal University of Campina Grande, Brazil. I started my PhD course in June, 2014 and I intend to conclude it by the end 2017.

I have worked with Design by Contract (DBC) since 2012 in my Master thesis. Currently, in my PhD I’m working on nonconformances categorization for Spec# programs. Next, I intend to work on development of an automatic approach to consider the conformance problem in the context of refactoring, contributing to use of Design by Contract methodology and to construction of reliable programs. Previously, in my Master’s degree, I proposed conformance
checking of Java and JML (Java Modeling Language), in which we analyzed and categorized the main likely causes for nonconformances between Java programs and their JML contracts. Furthermore, based on this categorization, we proposed a number of heuristics to assign likely causes for nonconformances. Finally, a conformance checking tool named JMLOK\(^1\) was also improved to support programmers on fixing nonconformances. During my Master’s degree I have published some papers about my research:


\(^1\) http://massoni.computacao.ufcg.edu.br/home/jmlok
Evolving C Program Families Safely

Flávio Medeiros
Federal University of Campina Grande, Campina Grande, PB, Brazil
flaviomedeiros@copin.ufcg.edu.br

Abstract

The C preprocessor is a simple and effective tool. It is widely used to handle variability and solve portability problems in program families. In this context, developers normally use tools like GCC and Clang. However, these tools are not variability-aware, i.e., they preprocess the code and consider each configuration individually. As a result, even well known and widely used families, such as Linux and Apache, contain bad smells and bugs related to variability. To minimize this problem, we proposed an approach to safely evolve program families in C. We developed a strategy to detect bugs and defined refactorings to remove bad smells in preprocessor directives. Our supporting tool—named Colligens—implements our strategy to detect bugs and applies our refactorings automatically. By using our approach in 40 open-source C program families, we detect 121 bugs related to variability including syntax errors, memory and resource leaks, uninitialized variables, and dereference of null pointers. We submitted fixes and developers accepted 78% of the patches. We consider families of different domains, such as web servers, databases, text editors, and games, and they have distinct sizes, ranging from 4.9 thousand to 1.5 million lines of code. Also, we removed 477 bad smells in 12 C program families without cloning the source code as refactorings proposed in previous studies. Our work presents findings that are useful for C program family developers during their development tasks, contributing to minimize chances of introducing bugs related to variability and bad smells in preprocessor directives. As future work we are validating our refactorings with real developers and using their feedback to improve our strategy to detect bugs.

Biograph

I started my PhD at Federal University of Campina Grande (UFCG) in March 2012. I’m a member of the Software Productivity Group (SPG) and I work under supervision of professors Rohit Gheyi from Federal University of Campina Grande (UFCG) and Márcio Ribeiro from Federal University of Alagoas (UFAL). Currently I’m doing a PhD internship at Carnegie Mellon University (CMU) under supervision of professor Christian Kästner. I’m planning to finish my PhD in March 2016.

Main publications

Characterizing Product Line Architecture Recovery

Crescencio Rodrigues Lima Neto$^{1,2}$

$^1$ Computer Science Department - Federal University of Bahia (DCC/UFBA)
$^2$ Federal Institute of Bahia (IFBA)

Abstract. Over the years, the interest on Product Line Architecture (PLA) has increased, due to SPL inherent complexity. In this context, our study aims to investigate how software architecture recovery processes recognize and represent the variability. Moreover, we intend to indicate the most suitable architecture recovery process according to SPL approaches (proactive, extractive, and reactive). We performed a systematic mapping on PLA, in which a set of 93 studies was accounted. We also conducted a survey with 147 industrial participants, analyzing their current problems with software architecture and future wishes. Finally, we realized a focus group with 29 participants. On the one hand, the mapping helped to define the research problem. Most studies presented solution proposals to address variability-related issues in PLAs. Results indicate intense research activity in the area. On the other hand, the survey investigated industry problems related with software architecture. We identified stakeholders need updated architecture documents with right information. Furthermore, the focus groups were useful to foster insights of possible proposals. Participants demanded guidance during the PLA design and suggested the use of tools and examples to assist them. Since PLA research area is too broad, we decided to reduce the scope by focusing on PLA recovery. Therefore, we propose an approach to suggest architecture recovery process for SPL projects. Moreover, we plan to execute activities such as: (i) systematic literature review to identify the most used architecture recovery processes; (ii) survey expert opinion to calibrate the initial proposal, and finally; (iii) empirical studies in SPL projects to evaluate the final approach.

Short Bio: In 2013, Lima Neto started the PhD research at the Federal University of Bahia (UFBA). He is a member of the Reuse in Software Engineering (RiSE) and Software Design and Evolution (aSide) research groups. Crescencio has experience in the field of software engineering, with emphasis on methods, processes and design techniques for software product lines.

PhD end date: 2017
Classifying Metrics For Assessing Object-Oriented Software Maintainability: A Family Of Metrics’ Catalogs

Juliana Saraiva

Federal University of Paraiba
Rua da Mangueira, s/n, - Companhia de Tecidos Rio Tinto CEP 58297-000 - Rio Tinto - PB - Brasil
{julianajags}@dce.ufpb.br
http://www.dcx.ufpb.br/pessoas/juliana/inicio

Abstract. Object-Oriented Programming is one of the most used paradigms, and software maintainability is considered an important attribute in quality level. Thus, the Object-Oriented Software Maintainability (OOSM) has been studied through years, and many researchers have proposed a large number of metrics to measure it. As a consequence of the number and diversity of metrics, beyond the no standardization in metrics definition and naming, the decision-making process about which metrics can be adopted in OOSM evaluations is a difficult task. Therefore, a systematic mapping study was conducted in order to find which metrics are used as indicators in OOSM analyses. 138 studies were selected, resulting in 568 metrics found. Analyzing the 568 metrics, inconsistencies in metrics naming were found (231 metrics involved). Moreover, a metrics categorization has been proposed, and 7 categories and 17 subcategories were identified. These categories represent the adoption scenarios of OOSM metrics. A metrics web portal was developed to provide information about these metrics, and to generate metrics catalogs throughout the categories choice. Other researchers can also systematically feed this portal. Consequently, the results of this work can be the first steps towards metrics standardization. A quasi-experiment was conducted to check the coverage index of the catalogs generated using our approach over the catalogs suggested by experts. 90% of coverage was obtained. Also, a survey was conducted to check the experts opinion about the catalog generated. Thus, the coverage evaluation can be the first evidences of the usefulness of the proposed approach for metrics choice in OOSM evaluation.

Key words: Software Maintainability, Metrics, Object-oriented Software, Experimental Software Engineering

1 Biography

I received the B.S. degree in Computer Science from the Federal University of Paraiba (2007), M.S. degree in Computer Engineering from the State University
of Pernambuco (2010), and Ph.D. degree in Computer Science from the Federal University of Pernambuco (2014). Since 2010 to 2014, I was Assistant Professor at the Federal Rural University of Pernambuco in Brazil and nowadays, I am Assistant Professor at Federal University of Paraiba (http://www.dcx.ufpb.br/).

I have experience in Computer Science with emphasis on Software Engineering, and more specifically on Software Maintenance and Software Measurement. My research interests are: Software Maintenance and Evolution, Software Quality, Software Metrics, Programming Languages, Empirical Software Engineering.

I completed my thesis on February of 2014, working with software maintainability. I proposed an Object-Oriented Software Maintainability (OOSM) metrics' portal to gather all information about this type of metrics, and also provide metrics' catalogs generation by a framework contained in the portal. The catalogs' generation is based on the user researchs context, in which is represented by metrics categorization, such as tool support, environment of metrics using, software internal attributes associated to a metric, and so on. Recently, I got the 4th place on Student Research Competition at the 35th International Conference on Software Engineering (SRC/ICSE) with my thesis' research.

Complementarily, I have been working with research methods for Empirical Software Engineering, such as systematic mapping study and quasi-experiments. In addition, during my Ph.D. program, I was selected to attend the Microsoft Internship Program at Microsoft Research Lab in Redmond/WA, where I worked with Empirical Software Engineering Group (ESE) focused in software naturalness, using n-grams approach.
Privacy and Security Policies for Collaborative Crosscutting Tasks

Abstract. In collaborative software development, members contribute to projects by executing assigned tasks and submitting their contribution to code bases. It allows the execution of a number of tasks by different developers in independent and parallel way. However, naive, careless, or malicious developers might violate privacy and security constraints more easily than in centralized software development. Generally, checking the accordance to these constraints is hard because the code that developers submit could be crosscutting when changes necessary for executing tasks are scattered across classes. To avoid constraints violations in such scenarios, we propose the Salvum language to provide a common means of specifying authorization policies to be obeyed by the collaborative developers. Therefore, we expect to increase software quality and development productivity by allowing reduction of privacy and security violations as well as decrease of the code to be manually reviewed.

1 Short Bio

This section explains a little bit of my life and what I have done.

1.1 Personal Information

My name is Rodrigo Cardoso Amaral de Andrade, I live at Recife, Pernambuco, but I was born in 1987 on the 20th of January at Maceió, Alagoas.

1.2 Education

Nowadays I am a Phd student in Computer Science at Federal University of Pernambuco, Recife, Brazil. I am advised by Professor Paulo Henrique Monteiro Borba. The Phd started on March of 2013 and it is expected to end on March of 2017. My Phd work is sponsored by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). I obtained my master degree in 2012 and I undergraduate in Computer Science on the end of 2009, both at Federal University of Pernambuco and advised by Paulo Henrique Monteiro Borba. Additionally I took Java Web and C# courses at Qualiti and Microsoft Innovation Center respectively.
1.3 Publications

Publications I was involved as author are in the following:

**Journal**


**Conferences and Events**


1.4 Other activities

I had an internship at Hive.log and E.Life where I worked as a programmer. The former is a company that provides software solutions for logistics whereas the latter provides software for user opinion analysis researches. Besides that, I had been a professor assistant for several courses which involves Data base, Software Engineering, and Programming languages.
Characterizing Code Smell Detection

José Amancio Macedo Santos

Federal University of Bahia
Computer Science Graduate Program
Software Engineering Laboratory

Context: “Code smells” are commonly presented as indicatives of problems in the design of software systems. However, some empirical studies have presented findings in the opposite way. The subject is not well understood. The area lacks information on which variables affect the human perception of code smells. **Objective:** Our research aims to explore the reasons of these divergent findings. More specifically, we propose to characterize Code Smell Detection (CSD) with respect to a set of independent variables. **Method:** We face the problem from two main fronts. First, we propose a family of controlled experiments to address cause-effect relationships between a set of characterization variables and CSD agreement among human subjects. Second, we propose a literature thematic synthesis of the knowledge on the topic. **Work performed so far:** We have performed four controlled experiments. The experiments explore how a set of independent variables (software visualization support, individual skill, experience, software domain and training) impact on CSD agreement, as well as the CSD strategies and decision drivers adopted by the participants. The synthesis, involving selection, coding and thematic classification, is in its middle stage and did not produce any conclusive findings yet. **Partial results:** Patterns of CSD strategies were found among the participants. Among the variables analyzed, software visualization support is the only one that does not impact CSD agreement. **Conclusion:** Our evidences show that CSD is more related to human traits than to the use of tool support. We are now trying to better understand this issue. **Keywords:** Code smell; controlled experiment; thematic synthesis.

José Amancio Macedo Santos started the PhD course in 2011, at Federal University of Bahia, Brazil. The end date is March, 2015. He is a college professor since 2006, at State University of Feira de Santana, Brazil. In 2003, he received his master degree from Federal University of Campina Grande, Brazil.
Parallel Development of Tasks

Paola Accioly

Federal University of Pernambuco,
Recife, Brazil
{prga}@cin.ufpe.br
http://www.cin.ufpe.br

Abstract. Software modularization has been long used as a mechanism to shorten the development time. In theory, the more modular a software is, the easier it is to develop and maintain because modules work as independent units and separate teams can develop them with little need for communication [1]. However, the different nature of development tasks (creation of a new feature, refactoring, bug fix, etc.) do not always match with the underlying modular structure. Tasks can frequently crosscut different modules leading to integration conflicts when two or more teams develop tasks separately [2]. There is evidence that conflicting changes in parallel development occur frequently and are rather a norm than the exception [3–5]. Such conflicts impact both development productivity—since identifying and fixing them is costly—and software quality—when defects escape. Despite the existing evidence there is still a lot to learn about these conflicts. In particular, the existing studies that bring evidence on conflicts occurrence use the line based merge, which merges the code based on textual similarity. Instead, there is a strategy called semistructured merge [6] that provides structural information about the code artifacts, so that it solves some conflicts automatically, reducing significantly their occurrence. We derived a catalog of conflict patterns that the semistructured merge identifies and we are now empirically assessing their occurrences in open source projects from GitHub to answer how frequently they occur, what are the consequences and what factors could cause an increase of conflicts. Answering these questions can help us to provide a better support to programmers by, for example, identifying and avoiding conflicts during the parallel tasks development.

Keywords: Software Modularity, Parallel Development, Integration Conflicts, Empirical Studies.
1 Short Bio

Paola Accioly is currently a Computer Science PhD Student since March 2013 with possible end date in March 2017 at the Informatics Center of the Federal University of Pernambuco, Brazil. Her research interests are focused in the following areas: software modularity, collaborative development, software product lines and empirical software engineering.

References

A new approach to bolster Software Engineering Learning Curve

Troy C Kohwalter
Instituto de Computação, Universidade Federal Fluminense, Niterói – RJ, Brazil
tkohwalter@ic.uff.br

Abstract. Software engineering is focused on practical and theoretical aspects of the software production. Teaching software engineering is traditionally done through theoretical classes with some practical exercises. Recently, games and simulators were introduced as a ludic alternative for software engineering learning, where decisions and interactions become key factors to transmit and acquire knowledge. However, mistakes made by wrong decisions may jeopardize the learning process, especially when reproducing its effects is not a viable option due to the non-deterministic nature of games. Therefore, an analysis and understanding of events, mistakes, and flows of a concrete game play may be useful for understanding educational and learning aspects in serious games. With this in mind, we proposed a novel approach called Provenance in Games, which monitors the game session and collects provenance. This provenance encodes the cause-effect relations of the game flow and can be further explored by queries or visualizations, providing the means for a post-game analysis. As benefits of our approach, we envision the improvement of the player’s understanding of the game session, providing insights on how the story progressed and how his decisions and events influenced the outcome. Moreover, we see our approach as an important instrument for tutors, as they can identify aspects that were not assimilated by students. Finally, we intent to run data mining and diff techniques over the provenance of multiple sessions, allowing the identification of communalities and differences among the students decisions and results.

1 Biography

Troy Kohwalter is a doctoral student at the Fluminense Federal University (UFF). He started his PhD in 2013 and is scheduled to finish it in 2017. He published three papers related to his thesis in conferences about games (SBGAMES2012, ICEC2013, and ACE2013) and software engineering (SBES2014).
An Approach for Analyzing Architectures Using Architectural Properties in SysML

Daniel Cunha da Silva
Federal University of Rio Grande do Norte - UFRN
Center of Exact Sciences and Earth - CCET
Department of Informatics and Applied Mathematics – DIMAp
(+55 84) 3215-3813 – 59072-970 – Natal/RN
danielcsilval@yahoo.com.br

Abstract. Software architectures design decisions are driven primarily by functional and non-functional requirements. Those decisions have a significant impact on the overall structure and behavior of the system and must be in conformance with the requirements. The aim of this paper is to verify whether it is possible to identify relationships between the architectural properties and the non-functional requirements that make possible to analyze how changes in architectural properties may affect these requirements. To analyze the architectural properties in the architecture, we need a modeling language. In our approach we use SysML, a general purpose language that extends UML. We first need to verify if SysML is actually able to help the architect in the representation of the architectural properties. As a case study, we are modeling web system architectures that uses the JEE technology. We need to decide which architectural properties can be observed and how we identify the impact and relationships with system requirements. Examples of architectural properties are: number and type of interfaces, number of components, number and type of connections and number of ports. Our research is investigating if we can establish a correspondence rules to the requirements, such as performance, security and maintainability. The study aims to investigate that, analyzing SysML, how changes in architectural properties affect requirements of the system.

Keywords. Non-functional requirements, architectural properties, SysML

1 Short Bio

Daniel Cunha da Silva is a PhD student of the Postgraduate Program in Systems and Computer of Federal University of Rio Grande do Norte, oriented by professor Dr. Jair Cavalcanti Leite. Started the program through a selection process since February 2011, which has been researching heavily in the area of software architecture and architectural models using SysML. He has a target completion of his course in February 2015.
Developers Assignment for Merging Branches

Catarina Costa¹

¹Computing Institute – Fluminense Federal University, UFF.
Niterói - RJ, Brazil
catarina@ufac.br

Abstract. During the software development process, artifacts are constructed and manipulated by many developers working in parallel. A common practice in this process is the parallel development of the source code using branches. Usually, at some point, the merge of these branches may be necessary. This process can combine two independent and eventually long sequences of commits, which may have been performed by different developers. Conflicts resulting from the merge of parallel changes may arise. When these conflicts are not automatically solved by the version control system, the developers in charge of the merge process must act. However, the developers’ knowledge regarding the changes performed in parallel is usually not taken into consideration when assigning developers to the merge task. Thus, this work aims at proposing an approach for assigning participants to merge branches, helping in the decision to choose the most appropriate developer to perform the merge. The approach is based on the analysis of the repository history until the point that precedes the merge. The approach considers different artifact granularities for developer assignment: the project as a whole, files, and parts of files (i.e., methods) changed. Considering the project, the approach counts the number of commits owned by each developer, prioritizing developers that committed in both branches. Considering files, it is possible to identify which files were changed in both branches and who changed them. The version control system provides information about who has changed each file. By considering parts of the files, it is possible to identify which methods were affected by the changes and who changed these methods. This information can be obtained from Abstract Syntax Tree (AST) analysis.

Biography

Catarina Costa is a doctoral student at the Fluminense Federal University. She started her PhD in 2012 and expects to finish it in 2016. She published two papers related to her thesis at ICGSE2013 and SEKE2014.
Detecting Behavioral Changes in Refactorings of Concurrent Programs

Felipe Pontes
Department of Computing and Systems
Federal University of Campina Grande
Campina Grande, PB, 58429-900, Brazil
felipepontes@copin.ufcg.edu.br
http://www.dsc.ufcg.edu.br/~spg/felipepontes

Abstract. A concurrent program is thread-safe if it behaves correctly even when accessed from multiple threads. In this context, refactoring is the process of changing a software in such a way that it keeps the program thread-safe and improves its internal structure. Widely used programming tools help applying refactorings using static analysis to try to detect if the resulting code preserves behavior. Other tools, like SafeRefactor (SR), try to detect behavioral changes through automatic tests generation and execution. However, those tools do not ensure that a refactoring applied to a concurrent program maintains it thread-safe because they do not consider multiple threads. Furthermore, the parallel programming model is counterintuitive because it confronts the way humans think. For this reason, the application of refactorings in concurrent programs is more likely to introduce bugs than in sequential one. We propose ThreadSafeRefactor (TSR), a tool based on SR, to detect if a transformation keeps a concurrent program thread-safe. Differently from SR which generates tests using only one thread, TSR generates tests containing multiple threads and execute them in the programs before and after applying the transformation. If TSR detects different test results, it assumes that refactoring does not keep the program thread-safe. As future work we will evaluate TSR initially using Java transformations applied to programs which behavior is not preserved. Then we will adapt TSR to generate tests only for the entities impacted by a transformation. Next we will implement other versions of TSR changing tests generation module to use different approaches. Finally, we will propose a different notion of equivalence between two programs to compare results from those different versions of TSR.

Keywords: Refactoring, concurrent programs, thread-safe, tests

Short Bio

Felipe Pontes is a member of Software Productivity Group (SPG) and PhD Candidate at Department of Computing and Systems in Federal University of Campina Grande (UFCG) (started at March 2014). He got his MSc. in Computer
Science also from UFCG in 2008 where he studied Sybil attacks in BitTorrent protocol. In a Sybil attack, an entity associates multiple identities to itself in an attempt to fool the other peers that execute the agreed protocol and increase its utility. Felipe Pontes works on industry since 2008 and he has experience in developing enterprise (Java EE), mobile (Android) and Web (Grails, PHP and C#) applications. Currently he is also a member of Embedded Lab where manages a team which develops embedded applications for industry. A possibly date to finish PhD is March 2018.

Main Publications

Sustainable Software Evolution for Ecosystem Architectures

Simone da Silva Amorim

Department of Computer Science, Federal University of Bahia
Av. Adhemar de Barros, s/n, 40.170-110, Salvador - Bahia, Brazil
ssamorim@dcc.ufba.br

Abstract. A Software Ecosystem consists in a community that constructs relevant software solutions in the context of a shared market or a common technological platform. Architectures in this environment have emerged as a new research area promising competitive advantages. Based on these advantages, organizations have invested many resources into any ecosystem of which they are member. To realize an acceptable return on investment, ecosystem platform must address as many of their current product needs as possible and be usable over several years for future products. Several issues impact on this platform. External and internal developers, as well as political, economical and technical concerns can have an influence upon and are influenced by it. These interactions are so intertwined that they are difficult to measure and evaluate. In face of this scenario, software evolution of these architectures should consider not only technical aspects, but also consider business aspects and their relationships. Our Ph.D. research aims at developing an approach to improve ecosystem architectures evolution. The approach intends to map technical and business dependences and provide choices for determining which design decisions may be suitable to implementation mechanism. This way, our approach supports architects during system design and system evolution in order to improve essential quality attributes such as Scalability, Extensibility and Flexibility for ecosystem architectures. We plan to perform empirical evaluations in order to assess the proposal effectiveness.

Keywords. Software Evolution, Software Architecture, Software Ecosystems.

1 Short Bio

Simone da Silva Amorim is a Ph.D. student in the software engineering area at Federal University of Bahia, Brazil. She starts the Ph.D. course in March 2011 and intends to conclude in December 2016. She is member of aSide - Research Group in Software Design and Evolution and the RiSE Labs. Simone is investigating Software Ecosystem Architectures. Simone and her advisors are publishing together some initial studies about ecosystem architectures as part of her Ph.D. thesis. She is also assis-
tant professor at Federal Institute of Education, Science and Technology of Bahia, where she teaches Software Engineering and Information Systems.

During her Ph.D. course, Simone participated of SPLSmart – Developing a Software Product Line for Management Medical Applications research project, when she studied SPL architectures in practice. Moreover, she did Ph.D. sandwich with the SoftArch research group at Southern University of California. She also helped to organize some events such as 4th RiSE International Summer School - RISS 2011, 16th Software Product Lines Conference - SPLC 2012 and XII Simpósio Brasileiro de Qualidade de Software - SBQS 2013. Aside from organizing research events, she had opportunity to present a short Lecture in ArchClouds: Architecting for Cloud-enabled Software Systems at IT University of Copenhagen and she has some publications about ecosystems research at International Workshop on Ecosystem Architectures – WEA 2013 and 2014, and Working IEEE/IFIP Conference on Software Architecture – WICSA 2014.

Before joined to this university, Simone worked twelve years as Analyst Developer in different companies getting large practical experience. Simone got M.Sc. Computer Science in 2004 at State University of Campinas, São Paulo, Brazil. Besides, she worked in Center of Research and Development in Telecommunications (CPqD) as System Analyst where she had involved since the conception solutions until the deployment of Geographic Information Systems.
Continuous Software Development: Unifying development and production environments

Rodrigo A. Vilar
Departamento de Ciencias Exatas, UFPB - Campus IV
Rua da Mangueira, s/n, Rio Tinto, Brazil
rodrigovilar@dce.ufpb.br

Abstract. The Software Engineering community constantly improves its techniques in order to increase productivity and quality. In this work, we propose a new SE approach Continuous Software Development (CSD) removing the typical boundary between development and production environments. Wiki systems have achieved a similar innovation. Before them, website development had separated authoring and publishing environments. Wikis united author and end user interfaces in the same system. CSD proposes the same concept for software development. In addition to implementing common functionality, production systems would provide self-evolution operations. Due to performance issues, the current approaches that change functionality on running software confine evolution capability in systems parts (hot spots). We propose to implement CSD in whole system as like as Wiki sites do. We have set our research scope to 3-tier web information systems (GUI, Business Logic and Persistence). Our research questions are: Does CSD impact developers productivity and systems quality? Can CSD pull the client into software development process? Since production data is available, can CSD enhance software evolution tasks? Our methodology has four phases: (i) define a development process for CSD, which uses production data to enhance evolution tasks, e. g. show evolution previews to client, block changes that causes inconsistencies; (ii) design a self-evolution API for CSD systems, based on ideas, like Scaffolding, from modern web frameworks; (iii) develop a CSD platform for 3-tier web information systems using Ajax and Java. (iv) perform experiments with developers creating some systems with CSD, Rails and Grails. Compare the resulting metrics for productivity, quality and client cooperation.

Keywords: Productivity; Quality; Self-evolution;

1 Short Bio

Rodrigo Vilar is a professor in the Department of Exact Science at Federal University of Paraiba. Since 03/2013, he is also a PhD student in the Department of Computer Science at Federal University of Campina Grande (planning to conclude his PhD studies in 10/2017). His research interests include meta modeling, experimental software engineering and productivity. He holds a MSc in Computer Science from the Federal University of Campina Grande.
A contribution to the definition and automation of structural testing of autonomous vehicles

Vânia de Oliveira Neves

Depto de Sistemas de Computação - ICMC
Universidade de São Paulo - São Carlos, SP - Brasil
vaniaon@icmc.usp.br

Abstract. The software used to control autonomous vehicles is a type of embedded system that needs to undergo strenuous testing before deployment. Field testing is the final stage of testing which ensures that the autonomous vehicles show the intended behavior, but it usually does not take into consideration the code structure. The vehicle could pass the testing process, but important parts of the code may never have been executed. In this context, it is important apply structural testing techniques to test software that is part of an embedded system as it ensures that critical sections of code have been executed. Therefore, the main objective of this work is the investigation of structural testing for embedded systems.

A testing model and a software tool to support structural testing in the context of autonomous vehicle field testing have been proposed. The model and the tool are based on the assumption that field testing logs related to specific versions of the program and scenarios are captured and can eventually be visualized, analyzed and compared offline and/or online according to control flow criteria. Using the coverage obtained from field testing and the existing logs, the model and the tool also include automatic test data generation using genetic algorithms to derive new logs and, thus, increase the initial coverage.

Apart from the known advantages of applying structural testing, it is intended that this approach will also help the tester to create new scenarios for field testing. Moreover, with the online analysis, the tester can observe the autonomous vehicle while it is running, collect information on the commands exercised and show coverage information that can be used to change the program or the scenario dynamically to better reach the test objectives.

1 Short bio

Vânia de Oliveira Neves is a member of Laboratório de Engenharia de Software (LabES) [Software Engineering Lab] at ICMC-USP in São Carlos, where she is in the final year of her Ph.D. (forecast to finish in December). She was born in Limeira/São Paulo (Brazil) in 1983. In 2006 she has obtained his Bachelor degree in Computer Science from São Paulo State University (UNESP). In 2010 she obtained her Master degree from University of São Paulo in São Carlos with a thesis.
titled “Contextual integration testing of object and aspect-oriented programs: criteria and automation” (advisor: Prof. Paulo Cesar Masiero). On August 2010 she joined the Ph.D. program also at University of São Paulo in São Carlos (advisors: Prof. Paulo Cesar Masiero and Prof. Marcio Eduardo Delamaro). Her research project relates to the structural testing of embedded software systems, in the context of projects in development at the Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos (INCT-SEC) [National Science and Technology Institute for Critical Embedded Systems], in particular in the field of autonomous mobile robots. The projects also involve investigative techniques and approaches for generating test data that can be applied in this domain. This step was performed as a Visiting Student Researcher at the Politecnico di Milano, in Milan from September 2012 to May 2013.

From December 2009 to September 2010 and from August 2004 to July 2006 she has been working as a software developer in a company that developed internet banking solutions for a large Brazilian bank and another company that developed ERP solutions. From March 2010 to December 2010 she has also taught courses at universities focusing on programming languages and design patterns, in particular Java.

Her research interests include software engineering, software testing, embedded software systems, object oriented programs, and aspect oriented programs.
Safe Evolution of Software Product Lines and Sets of Product Lines

Leopoldo Teixeira

Informatics Center, Federal University of Pernambuco
lmt@cin.ufpe.br

Abstract. Software Product Lines explore reuse among software systems, providing a systematic way to generate them. Extracting and evolving product lines can be costly and error-prone. Such tasks can benefit from refactorings with formal basis, to ensure correctness by construction. In this work, we propose a theory of product line refinement, which formalizes the notion of safe evolution in terms of a refinement relation over product lines. Refinement captures behavior preservation but abstracts quality improvement. The refinement theory is generic with respect to the different languages that we can use to implement product lines. Assumptions and axioms explicitly establish the interfaces between our theory and particular languages. Instantiating the theories with different combinations of concrete languages enables us to derive refinement templates that abstract safe evolution transformations. We investigate and explore similarities between these concrete languages, which ultimately result on templates that work at a higher abstraction level and can be reused for many languages. Thus, we propose a product line of theories, where different languages correspond to features, and products correspond to instantiations of the theories. We use the Prototype Verification System to encode and prove soundness of the theories and their instantiations. To evaluate expressiveness of the proposed templates, we analyze the evolution history of existing product lines. We observe whether templates are expressive to address safe evolution scenarios. Additionally, we also study the Linux kernel evolution and discover that most evolution patterns conform to the product line refinement notion. Finally, we also use refinement templates to reason about evolution of the product line of theories.

Keywords: Software Product Line, Software Evolution, Refinement
Biography

Leopoldo Teixeira is joining the faculty of the Informatics Center at the Federal University of Pernambuco, Brazil. He recently finished his Ph.D. at the same institution, supervised by professor Paulo Borba. During his graduate studies, he had the opportunity to spend a term visiting the Generative Software Development Lab at the University of Waterloo, under supervision of professor Krzysztof Czarnecki.

His main research interests are in the following topics and their integration: software product lines, refactoring, and formal methods. His thesis work focused on providing the formal foundations for reasoning about safe evolution of product lines and sets of product lines, which resulted in a set of refinement theories. These theories are the basis for deriving refinement templates, which are the basis for creating product line refactoring tools. He is also interested on how the theories and associated properties can help on designing refactoring checkers for product lines.
Cookbooks for Framework Instantiation: Semi-automated Construction Driven by Examples

Raquel Fialho Q. Lafetá

Faculty of Computing, Federal University of Uberlândia,
raquel.rafialho@gmail.com
WWW home page: http://lascam.facom.ufu.br

Software reuse is one of the major goals in software engineering. Object-oriented application frameworks promote the reuse of not only individual building blocks, but reuse of system design. However, framework instantiation requires a reasonable understanding effort. High-quality documentation is essential to minimize this problem. However, in most cases, appropriate documentation does not exist or is not updated. Our hypothesis is that the framework code itself and existing instantiations can serve as a guide for new instantiations. The challenge is that users still have to read large portions of code that hinders the understanding process, so our goal is to provide a semi-automatic approach to produce documentation from existing instantiations that could aid developers during the instantiation process.

Framework instantiation is an activity that is based on well-defined basic tasks, such as, subclassing. We propose a semi-automated approach using dynamic analysis, static analysis and design pattern detection to retrieve tasks information from the source code of the framework itself and existing instantiations. The final documentation should be presented in a cookbook-like document, where the recipes are composed of programming tasks and examples. Our approach distinguishes from previous works, because they are based either on information retrieval techniques, or on existing documentation, or on the knowledge of an experienced analyst.

A preliminary experiment will be conducted to verify that recipes address the requirements of a instantiation, showing that they could drive existing instantiations. To verify that the cookbook really reduces the comprehension effort during the instantiation activity, we intend to perform experiments with human subjects, on real instantiation activities. Currently, the process of reverse engineering has been defined and tested and is being automated with third-party tools. Further steps include the integration of different tools that automates the generation of the cookbook, and the detailed experimental planning.

Raquel Fialho de Queiroz Lafetá is a PhD student, started on March 2012, and end date planned to March 2016, at the Graduate Program in Computer Science from the Federal University of Uberlândia. She received her BSc (2007) and MSc (2011) degrees in Computer Science from the Federal University of Uberlândia. Her current research interest includes software analytics and program comprehension.
Evaluating and Redesigning Software Applications to Meet Usability Standards and Improve User eXperience Early in the Development Process

Luis Rivero
USES - Grupo de Usabilidade e Engenharia de Software
Programa de Pós-Graduação em Informática
Instituto de Computação – Universidade Federal do Amazonas
Manaus - Brazil
luisrivero@icomp.ufam.edu.br

Abstract. User eXperience (UX) is crucial for achieving software quality. UX explores how a person feels about using a software product and verifies if it meets usability standards, which check the ability of the software to be understood and operated. Meeting usability standards and achieving a high quality UX can allow increasing the adoption of a software product and the satisfaction of its end users. Although there are several methods for evaluating usability and UX in fully functional software, few of them can be applied at early phases of the development process. Also, there is a need for methods that can be applied by software engineers with low experience in usability and UX evaluations so that they can enhance the quality of their software products at a lower cost. To support software engineers in evaluating and redesigning pleasurable and easy to use software applications, this PhD research suggests combining the advantages of applying usability and UX evaluation methods in a unified process. First, we will provide a set of technologies for supporting the identification of usability problems early in development. Then, through the application of specific proposed UX evaluation methods for software engineers, we intend to assist the problems’ correction prioritization process based on how they affect the users’ emotions. After that, we intend to provide ways of suggesting possible solutions for improving usability and UX at the same time. This is an ongoing research and we are currently performing a systematic literature review to identify usability and UX methods than can be improved to meet this research’s goals. Furthermore, we are currently performing empirical studies aiming at evaluating the impact of correcting usability problems in the emotion of users.

Keywords: Usability, User eXperience, Software Quality, Early Development Phases, Evaluation Method, Redesign Method

1 Short Biography

MSc. Luis Rivero is a researcher from the Usability and Software Engineering Group (USES) at Manaus – Brazil, who focuses on developing solutions for improving soft-
ware quality in terms of usability and user experience. Luis finished his master thesis at Universidade Federal do Amazonas in March 2013 regarding usability inspection methods of Web applications. During his MSc. research, Luis developed a set of technologies for inspecting the usability of Web applications earlier in the development process to reduce problem correction costs. For his research, Luis has been awarded the prize of the “Best Work” at the Brazilian Symposium on Software Engineering [1] (SBES 2012) and one of his papers has been selected as one of the best papers at the Empirical Software Engineering Latin American Workshop [2] (ESELAW 2012). Also, his master thesis was considered one of the best theses regarding software quality at the Software Quality Theses and Dissertations Contest from the Brazilian Symposium on Software Quality (SBQS 2014). Luis is currently enrolled in the Informatics PhD course at Universidade Federal do Amazonas. His PhD course started on March 2013 and will probably finish on February 2017.

References

An Approach for Knowledge Management to Facilitate Improvements in Software Organizations

Jacilane Rabelo
Federal University of Amazonas – UFAM
jaci.rabelo@icomp.ufam.edu.br

Abstract. Software development activities are usually intense in knowledge, and they require the existence of activities for Knowledge Management (KM) related to the entire knowledge creation cycle. For the majority of software companies, the most valuable knowledge is kept in peoples’ minds, and this knowledge requires different methods to be managed. The knowledge that a software organization can hold and its ability to learn and use such knowledge are the main abilities of software organizations to maintain a competitive advantage and innovate. KM is essential to promote improvements and innovations in the software development processes. In that context, this research aims at investigating technologies (processes, techniques, methods, approaches, frameworks, practices and tools) to support the consensus on the value-based knowledge within a software organization and its use. This investigation will analyze the factors that positively and negatively influence KM in Software Engineering. The influencing factors will be associated with: a) the type of KM learning process (individuals, teams, organizational); b) knowledge area(s) within Software Engineering; c) type of software development (agile, traditional); d) organization’s size; e) focus on improving the software process; and f) stage of the knowledge cycle. A new approach will be developed based on the findings of this research and using a specific theory of knowledge creation. The research methodology for the development of the approach is based on: a systematic literature review, semi-structured interviews with software organizations and empirical studies to ensure adequate assistance to the software industry. Currently, this research is in its review phase, extracting KM influencing factors found on related researches. We expect that by proposing such approach, we will support the improvement of knowledge management practices, enabling common understanding of the value of knowledge in software organizations.

Keywords: knowledge management; knowledge value; sustainable knowledge management improvement; software engineering.

Short Bio: This research is being conducted by Jacilane de Holanda Rabelo, a PhD student from the Postgraduate Program in Informatics (PPGI) at Federal University of Amazonas (UFAM). Jacilane holds a master’s degree in Computer Science with a focus on Software Engineering at the same university. Her
admission in the doctoral program was in March 2013, and the course completion is expected on February 2017. Jacilane has experience in Knowledge Management in Software Engineering and has published some works related to such theme: a) Using PABC-Pattern for Knowledge Codification: An Empirical Study, 2014, Brazilian Symposium of Software Quality (SBQS); b) A Qualitative Study about the Life Cycle of Lessons Learned, 2013, International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE 2013); c) Monk: A Tool Support for Knowledge Socialization in Software Teams, 2013, Brazilian Symposium of Collaborative Systems (SBSC); and d) Comparing Knowledge Codification Approaches: An Empirical Study, 2012, Brazilian Symposium of Collaborative Systems (SBSC). Finally, Jacilane has experience in research projects with topics related to the research area of Knowledge management in software engineering.
On the Reconstruction of High-Level Software Abstractions from Execution Traces

Liliane do Nascimento Vale\textsuperscript{1,2}

\textsuperscript{1} Faculty of Computing - Federal University of Uberlândia, Uberlândia – MG – Brazil  
\textsuperscript{2} Computer Science Department - Federal University of Goiás, Catalão – GO – Brazil  
liliane.ufg@gmail.com

An important issue in software evolution is the effort needed to understand existing applications. Reverse engineering techniques play an important role for reducing this effort. Unfortunately, there is no widely adopted way to extract and visualize high-level abstractions from source code. Our hypothesis is that a small set of interesting (i.e. with a high-level meaning) code elements – classes and methods – can provide a coherent explanation of software feature’s internals. We proposed a new technique to reconstruct high-level abstractions of software in terms of their features using dynamic analysis. High-level abstractions are characterized as a very summarized sequence of method calls directly mapped from use case actions. In the proposed approach, firstly, the software features of interest are manually identified and represented as use cases, which are characterized as a set of actions. Execution traces are extracted and stored, accordingly to the defined use cases, and then compressed to remove loops and recursion. Next, method calls near to the leaves of the trace call tree are considered the least interesting and filtered out. The remaining elements are organized in a colored visualization of their method call tree roots in order to identify shared and distinct code elements of features. To retrieve even more relevant information corresponding to an action of a use case, a query processor customized with the terms extracted from code elements is used iteratively to select the final relevant code elements. The main contribution of this approach is a novel summarization technique of execution traces that enables developers that have no (or very little) knowledge on the target system to capture code elements with a high-level meaning and reconstruct a coherent high-level abstraction that explain how the system’s features are implemented. Currently, we have conducted a qualitative study with ArgoUML and Tomcat that has demonstrated the extent of effectiveness and some limitations of the technique. Further improvements and more extensive evaluation have still to be designed.

Liliane do Nascimento Vale is a PhD student at the Graduate Program in Computer Science from the Federal University of Uberlândia. She started her PhD course in 2013 with conclusion planned for 2016. She received her BSc (2006) degree in Computer Science from the Federal University of Goiás and MSc (2009) degree in Computer Science from the Federal University of
Uberlândia. She is an Assistant Professor at the Computer Science Department of the Federal University of Goiás (UFG). Her current research interests includes software analytics and program comprehension.
Conquering Exceptions with a Recommender System

Eiji Adachi Barbosa
Opus Research Group, Informatics Department, PUC-Rio, Brasil
ebarbosa@inf.puc-rio.br

Abstract. Despite their wide adoption in popular programming languages, exception handling mechanisms (EHMs) are not properly used since many developers still ignore exceptions. Without support, developers struggle to implement exception handling. The lack of an explicit exception handling policy in most software projects is one of the factors that hinder exception handling implementation. Exception handling policies refer to software designers’ intent on how and where exceptions should be used and handled in a software project. In fact, the poor implementation of exception handling often introduces faults in the source code and many of these faults (e.g. uncaught exceptions) are related to the lack of an explicit and verifiable policy. Current EHMs implemented by most programming languages, as well as previous solutions aimed at supporting developers in using exceptions, provide limited support to specify and verify exception handling policies. To fill this gap, we proposed a domain-specific language (DSL) to specify and verify exception handling rules that the source code must adhere. We performed a user-centric evaluation with developers and observed that participants exhibited a positive attitude towards language adoption. More importantly, results from our quantitative assessments with industrial systems suggest that identifying violations of exception handling policies may assist in discovering faults in the source code. Moreover, we are designing a recommender system to aid developers in correcting violations of exception handling policies in the source code. The proposed recommender system will use the source code and the exception handling policy as its knowledge base to recommend changes in the source code in order to solve detected violations.

Keywords. Exceptions, Exception handling, Recommender systems.

1 Short bio

Since March 2012, Eiji Adachi Barbosa is a PhD candidate at Pontifical Catholic University of Rio de Janeiro (PUC-Rio) working under the guidance of Prof. Alessandro Garcia. His PhD research explores the intersection of recommender systems for software engineering and software robustness. In particular, his interests are focused on how recommender systems for software engineering can assist developers in producing better exception handling code in order to achieve higher software robustness. Eiji Adachi Barbosa received a bachelor’s degree in Computer Science in 2009 from Federal University of Rio Grande do Norte (under Prof. Thais Batista) and a master’s degree in Informatics in 2012 from PUC-Rio (under Prof. Alessandro Gar-
cia). Currently, he is on the third year of his PhD and he is expected to defend his thesis until March 2016.
Identification and Management of Technical Debt using Software Visualization Resources

Thiago S. Mendes$^{1,2}$

1 Fraunhofer Project Center for Software and System Engineering at UFBA, Salvador, Brazil
2 Federal Institute of Bahia – IFBA, Salvador, Brazil
thiagomendes@dcc.ufba.br

Abstract. The quality of software under maintenance often decreases over time. This is especially true when considering aspects such as its internal structure, adherence to standards, and documentation. One reason is that maintenance activities are often carried out under strong restrictions of time and resources. Deal with this question considering the metaphor of Technical Debt (TD) has helped some professionals to discuss software maintenance issues. The concept of TD illustrates the problem of maintenance tasks pending as a type of debt that brings a short-term benefit to the project, but that may have to be paid with interest later in the development process. It is common that software projects incur debts during its development process, since small amounts of debt can increase productivity. On the other hand, the presence of the debt brings risks to the project. Then, it is important to manage it. An important step for an effective management of the debt is its identification and monitoring.

My research aims to investigate new approaches for extracting metrics and visualization software to aid developers in tasks of identification and management of TD. The first contribution of this work was performing of systematic mapping in the literature to investigate which TD types and existing TD indicators, management strategies, data sources, artifacts, visual metaphors that have been used to identify and manage technical debt. With the findings, we created an Ontology of Terms on Technical Debt and a tool VisMiner (https://github.com/visminer/).

Now we are creating metrics and visual metaphors in the context of TD.

Keywords: software comprehension, technical debt, software visualization and software quality.

1 Short Bio

Mendes is a PhD student in Computer Science at Federal University of Bahia (UFBA/Brazil) (initial 2012 – March of 2016 (possibly tentative) end date) and assistant professor at Federal Institute of Education, Science and Technology of Bahia (IFBA). He is also a member of the Software Engineering Lab (LES/UFBA) and Applied Informatics Group (GIA/IFBA). He holds a MS (2008) in Computer Science from the Federal University of Viçosa (UFV/Brazil), a BA (2005) PhD in Computer Science from the University of Vale do Rio Doce (UNIVALE/Brazil).
Bug Reopening: Impact of Process and Design Variables

Rodrigo Souza
Federal University of Bahia
rodrigo@dcc.ufba.br

Abstract. Fixing bugs is an important software development task. In a typical scenario, a bug fix is created, tested, and integrated into a software release. Sometimes, however, someone discovers that the fix was incomplete or inappropriate. In this case, the corresponding bug report is reopened so it can be further investigated.

Reopened bugs cause rework, since they require new bug fixes to be written, tested, and integrated. Also, high bug reopening rates may indicate instability in the process or in the code base [1].

There are two types of bug reopening: triage- and fix-related. A triage-related reopening occurs when a bug report is initially closed (e.g., because someone thinks it is invalid), but then reopened after more information becomes available. On the other hand, a fix-related reopening occurs when a patch is accepted as a bug fix, but later is found to be inappropriate.

Recent studies have found the most common causes of bug reopening [1] and factors that influence the reopening rate [2–4]. They do not make a distinction, though, between triage- and fix-related reopening.

Our objective is to empirically assess the influence of design and process variables in the occurrence and impact of fix-related bug reopening. To this end, we perform statistical analyses on bug reports and source code history from open source projects.

So far, we have found two variables that appear to influence the reopening rate: release cycle length and presence of architectural violations. We discovered that, after Mozilla adopted a rapid release model, the reopening rate of its main product, Firefox, increased. We also found that, in Eclipse, bugs in classes that violate the prescribed architecture are more likely to be reopened.
Short Bio

Rodrigo Souza received a M.Sc. degree in Computer Science from the Federal University of Campina Grande (UFCG), Brazil, and, since March, 2010, is a Ph.D. student at the Federal University of Bahia (UFBA), Brazil (estimated end date: April, 2015). His main research interests are empirical software engineering, software evolution, and release engineering, with publications at the European Conference on Software Maintenance and Reengineering (CSMR) [5], at the Working Conference on Mining Software Repositories (MSR) [6], and at a workshop on the International Conference on Software Engineering (ICSE) [7, 8]. Rodrigo is also chief technology officer at the Office of Information Technology at Federal University of Bahia (STI-UFBA).

References

8. ——, “Patterns for extracting high level information from bug reports,” in Proc. of the 1st Workshop on Data Analysis Patterns in Softw. Engineering. IEEE, May 2013.
A Recommendation System for Product Configuration in Software Product Lines

Juliana Alves Pereira
jualvespereira@gmail.com

Abstract. Software product line (SPL) is a set of software systems that share a common and variable set of features, satisfying the specific needs of a particular market segment. A feature represents an increment in functionality relevant to some stakeholders. SPLs commonly use a feature model to capture and document common and varying features. The key challenge of using feature models is to derive a product configuration that satisfies customer preferences. Although automated support for product configuration has already been extensively investigated in the literature, some customer preferences are usually neglected. Therefore, this project proposes a recommendation system to support the problem of product configuration providing recommendation of features that increases customer satisfaction. Recommendation systems are software tools that can assist the user to find information and to make decisions. They are primarily directed towards users who lack sufficient personal experience or competence to evaluate the potentially overwhelming number of alternative items that an SPL, for example, may offer. In this context, we will use a knowledge-based recommender system approach, where the customers’ preferences toward particular features of the SPL are analyzed and then rules of customer interest profiles are thus drawn in order to recommend customers features that have potential attraction for them. We are considering use fuzzy methods for the representation and subsequent construction of justifications and recommendation rules. A practical application of the proposal is, for instance, recommending feature for a product upgrade. The novelty of this research lays in the investigation of two relevant Software Engineering topics, namely recommendation systems and SPL configuration.

Keywords: Software Product Lines, Product Configuration, Recommendation System.

Short bio. Juliana starts the PhD program in October 2014. She received his Master degree in Software Engineering from Federal University of Minas Gerais (UFMG) in 2014. Juliana also holds a BSc degree (2012) in Information Systems from the Federal University of Lavras (UFLA). His research interests include software product lines, software reuse, recommendation systems, search-based software engineering, empirical software engineering, software quality, software maintenance, and software engineering education. In recent years, she has published and revised research papers in Software Engineering conferences at Brazil. Member of Brazilian Computer Society (SBC).

Website: http://dcc.ufmg.br/~juliana.pereira.
Certification-Based Development Methodology of Biomedical Signal Acquisition Systems

Álvaro Alvares de Carvalho César Sobrinho

Federal University of Campina Grande
Rua Aprigio Veloso, 882, Bairro Universitário
Campina Grande, Paraíba, Brazil

Abstract. Software-controlled medical devices can be used alone or in devices network. The number of connected devices and embedded software is related to the probability of failure and complexity of systems. Hence, it is necessary to evaluate medical systems before commercialization to avoid risks to the safety of patients. Regulatory agencies such as the US Food and Drug Administration (FDA) define requirements to approve medical devices and the software used to control devices. The requirements are usually defined by prescriptive standards in a process-based approach. The ISO 14971 standard for risk management is one of the requirements. However, the conformance with activities only during the development process is not enough to prove systems’ dependability. For example, FDA recorded 1.210 computer-related recalls between 2006 and 2011 (90.5 percent of Class II). In this context, there is also a product-based approach in which one can use techniques such as formal methods to verify quality properties. This work proposes a certification-based development methodology for non-invasive biomedical signal acquisition systems (Class II). The methodology implements the process and product-based approaches. Physicians use these systems to analyze the electrical activity from the patients’ body. Electrodes are used to connect a measurement device to the patient in order to isolate, amplify, and convert electrical signals. On the other hand, embedded software are used to perform verifications such as skin-electrode impedance and battery situation. Design problems in these systems can induce physicians to errors in the evaluation of patients. Two case studies about the Electrogastrography (EGG) and Electrocardiography (ECG) systems have been implemented to test the methodology.

Keywords: Non-Invasive Medical Systems; Formal Methods; Embedded Systems.
**Biography.** I have a B.Sc. degree in Systems Analysis from the CESMAC University Center (2010), a M.Sc. degree in Computational Modeling of Knowledge from the Federal University of Alagoas (2013), and I have been a Ph.D. degree candidate in Computer Science from the Federal University of Campina Grande since February, 2013. My expected time until graduation is February, 2017.
Using CBAR Concepts to Automate Test Oracles for TTS systems

Rafael A. P. Oliveira
rpaes@icmc.usp.br
Dept. of Computer Systems, University of São Paulo – (ICMC/USP) São Carlos, SP

Abstract

Systems with complex outputs (GUIs, Web applications, Text-to-speech (TTS), etc.) make the test automation a difficult job. Still, systems with complex are currently present in a plenty of contemporary applications. Regarding TTS systems, among these applications, we highlight functionalities such as, reading e-mails, social network updates, books or headlines to blind people, traveling directions, etc. However, quality problems (e.g. pronunciation and intonation) make the synthesized speech highly unnatural, limiting their mainstream adoption. Even so, there is still no known general method to define test oracles for such systems. In a testing environment, Test Oracles represent the mechanism to evaluate whether a system execution is correct or not. The literature describes several techniques useful for measure complex-output systems’ quality, but they mostly involve informal interpretations, and human intervention. This Ph.D. research aims to use test oracles to deal with this problem, extending the framework O-FIm/CO (Oracle for Images and Complex Outputs) that uses CBAR (Content-Based Audio Retrieval) concepts to automate test oracles. CBAR includes an extensive treatment of audio data segmentation, indexing and retrieval audio based on analysis of its content. Using pre-implemented feature extractors (image and audio), we aim to provide an effective automated test oracle technique, supporting the quality assessment of such systems. The two first major results are a Systematic Mapping and several feature extractors for TTS systems written in Brazilian Portuguese. We have conducted preliminary empirical studies that revealed our oracles can achieve up to 74% of accuracy. We expect this approach to reduce the human efforts of testing complex-output systems.
Author’s short bio

Rafael A. P. Oliveira is a Ph.D. candidate at University of São Paulo (USP/ICMC). He has received his M.Sc. in 2012 from the University of São Paulo. He is a visiting scholar at the University of Maryland (UMD, College Park, MD, USA) under the supervision of Atif Memon. His research focuses on alleviating human testing efforts, automated and systematic testing strategies, automated test oracles, GUI and TTS testing. In addition, his main interests are software testing, quality, and reliability. His main publications can be found on his personal site (http://www.labes.icmc.usp.br/ rpaes/) and on Lattes Platform (http://lattes.cnpq.br/0793753941171478). (PhD start: March/2012 – possible end date: April/2016)

Acknowledgments

Rafael is supported by FAPESP (Grant Numbers 2012/06474-1 and 2013/01775-6). He also thanks Prof. Marcio E. Delamaro, Prof. Fátima L.S. Nunes, and Atif Memon the advisors of this research.
On the Synthesis of
Architecturally-relevant Code Anomalies

Willian Oizumi
OPUS Research Group, Software Engineering Lab (LES) – Informatics Department
Pontifical Catholic University of Rio de Janeiro (PUC-Rio) – Rio de Janeiro – RJ – Brazil
wnoizumi@inf.puc-rio.br

Abstract. Code anomalies are even more harmful when they represent software architecture problems, so-called architecturally-relevant code anomalies (AA). Given time constraints, developers often need to focus on identifying these critical anomalies. However, each architectural problem is often reified by multiple anomalies in the program structure. Therefore, before removing an architectural problem, developers first need to understand which anomalies contribute to the architectural problem and how anomalies are related to each other. Existing techniques for anomaly detection fail to synthesize these properties of AA and, as a consequence, make it difficult to understand the architecture problem in the context of a program. Moreover, there is limited knowledge in the literature about requirements for the synthesis of AA. Aiming that goal, we performed a wide range of inquisitive empirical studies, which were required to investigate and reveal these synthesis requirements. The inquisitive studies were conducted based on recurring characteristics of AA found in 7 industry-strength software systems. For instance, they revealed that the synthesis must include “contextual information” about the location and evolution of AA in the program. This information must be derived from different sources, either related to a single version or multiple versions of a system, supporting a better understanding of the scope and emergence of an architectural problem. Based on all the study results, a synthesis technique was defined and implemented. The next steps of this research include qualitative evaluations of our synthesis technique with the involvement of different user’s profiles. These evaluations are also intended to provide further insights for this PhD proposal.

Short Bio

Willian Oizumi is a Master candidate in Informatics Department of PUC-Rio with completion scheduled to March 2015, when he plans to become a PhD candidate on the same topic of his MSc. Following this schedule, he will try to finish his PhD in 2019. He is member of the OPUS Research Group, under the supervision of Professor Alessandro Garcia. He received a BSc degree in Computer Science in 2012 from the State University of Maringa (UEM). His main research fields are Software Architecture, Code Quality and Empirical Software Engineering.
Supporting newcomers overcoming barriers to contribute to Open Source Software projects

Igor Steinmacher

Departamento de Computação – Universidade Tecnológica Federal do Paraná
Departamento de Matemática e Estatística – Universidade de São Paulo
igorfs@utfpr.edu.br

Abstract. While onboarding to open source software (OSS) projects, newcomers face many barriers that hinder their first contribution, leading in many cases to dropouts. Many projects leverage the contribution of outsiders and the sustainability of the project relies on retaining some of these newcomers. Many OSS projects do not pay attention to the barriers faced by their newcomers or do not know how to mitigate them. This study aims to identify barriers hindering newcomers’ first contribution to OSS projects. We conducted a qualitative study with data obtained from semi-structured interviews with 36 subjects from 12 different projects, including newcomers and experienced members. We used procedures of Grounded Theory (GT) – open and axial coding – to analyze the data. So far we have a conceptual model composed of 57 barriers, grouped into six different categories: cultural differences, newcomers’ characteristics, reception issues, orientation, technical hurdles, and documentation problems. We plan to build a tool that organizes information and tools provided by the project according to the model defined. Our goal with this tool is to evaluate whether the model can be used to make newcomers aware of the problems and solutions they will encounter, providing a set of signs and maps to guide newcomers and alert them. To evaluate the model and the tool we plan to conduct a controlled empirical study with qualitative outcomes using diaries. The study will be conducted with undergrad students, who are potential OSS newcomers, who will receive it as an assignment in an Open Source course.

Keywords: newcomers, open source, barriers, joining, contribution

Short Bio. I am is a PhD candidate at University of São Paulo, under supervision of Marco Aurélio Gerota. My current interest is on newcomers to Open Source Software projects, mainly on the how to support their first steps. I have some papers published on this research topic. I started my PhD in August, 2011 and aim to finish it by the end of 2014. Currently I am also a faculty at Federal University of Technology - Paraná, where I teach Software Engineering, Project Management and Open Source related courses. As a faculty I advise undergrad students and coordinate a project related to Open Source. More details can be found in my CV: http://bit.ly/SteinmacherLattes
How can visualization help developers improve exception handling design?

Benjamin Jakobus
Opus Research Group, Informatics Department, Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil, benjamin.jakobus@les.inf.puc-rio.br

Abstract. Every year, software systems increase in complexity and size. As the complexity of these systems increases, so does the potential for the occurrence of exceptions - that is, exceptional and unexpected events at runtime. Designing ways of dealing with exceptions is therefore very important, and is not a trivial task. For example, there are many types of exception dependencies, i.e. exceptions can be raised, caught, re-mapped, re-thrown and propagated, whose effects and implications the developer must reason with, and the path that each single exception takes can be very complex. As one of the consequences of this being a non-trivial task, many developers tend to pay little attention to exception handling, or even ignore it altogether. As a result, exception handling code evolves to be very messy and convoluted, which often leads to the introduction of additional faults. At this point, fixing the introduced faults or improving the system’s exception handling becomes difficult. To help developers devise and improve their exception handlers, we propose an application for visualizing a system’s existing handlers alongside the developer’s intentions as to how he would want them to be adopted. Developers will be able to specify their exception handling intentions and validate them to make sure that they do not conflict with each other. They will then be visualized along with handlers already present in the system, showing dependencies, and the violations that occur if their intentions were implemented. We will empirically investigate if the proposed software visualizations are effective to help developers to devise handlers, as well as making exception handling easier to express and understand. The proposed application will be evaluated using the IBM Connections Toolkit.

1 Biography

Jakobus graduated with a BSc in Computer Science from University College Cork, after which he co-founded a start-up. He returned to the university one year later and obtained an MSc in Advanced Computing from Imperial College London, after which he took up a position as a software engineer at IBM. He is currently pursuing a PhD at PUC-Rio (approx. completion date: 2017).

Keywords: visualization, exception handling, exceptions
Do the refactoring activities improve software maintainability?

Diego Cedrim

Informatics Department
Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil
dcgrego@inf.puc-rio.br

Abstract. Refactoring is a common practice in the process of software maintenance. It aims at improving the structure of a program, making it easier to understand and quicker to modify. Code smells in a program are used as an indication of structural problems that may be solved using refactoring. Therefore, refactorings are important to supporting the removal of code smells and, as a consequence, they are expected to improve maintainability. In fact, developers often associate refactoring with improved maintainability. They assume this practice is effectively used to remove code smells, and reduce maintainability problems. In this context, we pose the following question: Do refactoring activities improve software maintainability? This research aims at answering this question by analyzing refactorings across software version pairs. For each pair, we identify existing code smells and refactorings. For each refactoring, we investigate if the existing smells were removed or new ones were created. Our preliminary dataset consists in 43 refactorings in three projects. Considering these cases, a decay of maintainability was observed in 10 of them (23.2%). The decay was observed through an increase on the number of code smells in the refactored code. Maintainability was improved in just nine (20.9%) of the cases. The most common behavior in our sample is the one in which no impact was observed. This kind of behavior was observed 24 times (55.8%). Our next steps include: (i) increase the dataset and make statistical analyses in all results, and (ii) propose a recommender system to better assist frequent types of refactorings by informing programmers about the potential negative effects. We also intend to use qualitative and quantitative studies to shed light on our next research steps.

Short Bio

PhD student at Pontifical Catholic University of Rio de Janeiro since 2014 and member of the OPUS Research Group, under the supervision of Prof. Alessandro Garcia. MSc in Computer Science, Pontifical Catholic University of Rio de Janeiro (2009). BS in Computer Science, Federal University of Alagoas (2006). He is a professional software engineer since 2006 and was Professor during three years. Participated of teams of web software development, mobile applications, scientific computing and naval systems.
Abstract. Refactoring is the process of improving the program structure without changing its behavior. Even common, refactoring might be an error-prone activity if developers unconsciously alter external program behavior. If refactored code does not preserve behavior as intended, this practice will adversely affect software quality. Unfortunately, refactoring analysis is a rather complex task. For instance, it requires extract code changes, judging these changes as intended refactorings and investigate whether they preserved behavior. Moreover, each refactoring is interleaved with other programming activities, further increasing the chance of programmers introducing behavior changes. In this context, we are empirically studying whether refactorings preserve program behavior. In our initial study of 21 versions of 3 software projects in collaboration with other researchers, we found almost 70% of intended refactorings preserve behavior. However, a rate of 30% of refactorings that not preserve behavior seems very high. This analysis was supported by regression testing and other state-of-the-art techniques for refactoring analysis (e.g. SafeRefactorImpact). Aforementioned results lead us to believe refactorings should be more carefully applied. Based on that preliminary study, if future works confirm that refactorings are poorly applied, we intend to propose a search-based technique to enhance the support for behavior-preserving refactorings. The technique will provide and rank recommendations of refactorings that (i) do not alter external behavior and (ii) do not introduce further structural problems in the program. As this research proposal is only at an early stage (1º semester), we also intend to use qualitative and quantitative studies to shed light on our next research steps.
Short Bio

Leonardo da Silva Sousa is a postgraduate student in Informatics Department of PUC-Rio since 2014 and member of the OPUS Research Group, under the supervision of Prof. Alessandro Garcia. He received a BSc degree in Computer Science in 2012 at Federal University of Mato Grosso (UFMT) and MSc degree in Computer Science in early 2014 at Federal University of Goiás (UFG). His main research interests include: code refactoring, code smell detection, search-based software engineering, software architecture and software testing.
On Leveraging Interactive Detection of Code Anomalies

Danyllo Albuquerque
OPUS Research Group, Software Engineering Lab (LES) – Informatics Department
Pontifical Catholic University of Rio de Janeiro (PUC-Rio) – Rio de Janeiro – RJ – Brazil
dwalbuquerque@inf.puc-rio.br

Abstract. Background: Early detection of code anomalies helps developers on performing effective refactoring actions. However, detecting code anomalies is traditionally supported by non-interactive detection (NID) approaches – i.e. developers are warned about anomaly instances only lately. Recently, interactive detection (ID) was proposed in order to improve early detection and refactoring code anomalies. In this approach, developers receive ongoing information about anomalies instances while performing programming activities.

Problem Statement and Goal: Studies revealed NID leads to negative consequences on software quality and induces programmers to perform unnecessary or late refactoring actions. Although ID seems a promising research direction, there is no knowledge about their impact on code anomaly detection and refactoring actions. Thus, our research aims at: (i) better characterizing ID approach; (ii) performing empirical studies to gather evidence about the (dis)-advantages of using this approach; and (iii) proposing and evaluating a novel ID approach aiming at coping the limitations of existing approaches.

Research Activities: First, we are performing a systematic literature review aiming at characterizing code anomaly detection approaches. Second, we are conducting a controlled experiment in order to assess the impact of ID on effective anomaly detection and refactoring actions.

Preliminary Results: Our initial analysis suggested there are no rigorous criteria in the literature to distinguish ID and NID approaches. Moreover, experimental results pointed out ID can be used to support anomaly detection since developers tend to identify a high number of anomaly instances.

Initial Conclusion and Future Work: The amount of false negatives may increase by using ID, and hence, lead to perform unnecessary refactoring actions. Our studies suggest amount and nature of the information provided by existing ID approaches might cause this problem. Therefore, further studies are required aiming at: (i) propose and evaluate a novel ID approach; and (ii) perform a comparison analysis with existing approaches for assessing the effectiveness in anomaly detection and refactoring actions.

Short Bio

Danyllo Albuquerque is a postgraduate student in Informatics Department of PUC-Rio since 2013 with completion scheduled course in 2017. He already completed the courses and he is now working on his PhD thesis proposal. He received a BSc degree in Computer Science in 2010, and the MSc degree in Computer Science in early 2013 from the Federal University of Paraiba (UFPB). In addition, he is a post graduated by Estácio/IDEZ University in Web Development in 2013. As a member of OPUS Research Group, he is particularly concerned in investigating advanced techniques for supporting modular software development. His specific research interests are in the areas of software architecture, refactoring, code quality analysis, experimental software engineering, and software testing.
Understanding the Time Effectiveness of Pairs in Identifying Code Anomalies

Roberto F. Oliveira¹

¹Software Engineering Lab (LES) – Informatics Department
Pontifical Catholic University of Rio de Janeiro (PUC-Rio)
Rio de Janeiro – RJ – Brazil
rfelicio@inf.puc-rio.br

Abstract. Background: A code anomaly is a program structure indicating possibly critical problems of software maintainability. Prevention and detection of code anomalies are time-consuming and cumbersome activities when performed by individual programmers. Therefore, the software industry is increasingly adopting pair programming – i.e. two people working together to produce and edit the code – in order to improve their software maintainability. In general, pair programming is often expected to have a significant impact on the prevention and detection of code anomalies. Therefore, there is a need to understand the impact of pairs on these critical activities. Research Activities: This research aims at: (i) characterizing different pair-wise approaches to the activities of code anomaly prevention and detection, (ii) evaluating the (dis)advantages of these approaches, and (iii) proposing a novel pair-wise approach for improving time effectiveness on preventing and detecting code anomalies. Initially, we performed a literature review about the adoption of pairs on software programming activities in general. We also conducted a first controlled experiment with novice programmers; the goal was to compare the time effectiveness of novice individuals and pairs in preventing or detecting code anomalies. Preliminary Results: The literature review only pointed out potential advantages on the effectiveness of pairs performing both activities. However, we observed contradicting and complementary (dis)advantages in the aforementioned controlled experiment. For instance, the amount of critical code anomalies identified by novice pairs is usually much higher in contrast to novice individuals. However, pairs did not reduce the amount of false positives for some types of code anomalies, there by leading developers to waste significant time performing in effective refactoring activities. Conclusion and Future Work: The initial results allow us to confirm the feasibility of using pairs in detecting some types of code anomalies. Aiming to improve our understanding about pair-wise approaches for preventing and detecting code anomalies, we plan to: (i) perform quantitative and qualitative studies with professional developers, and (ii) replicate the study where developers are also supported by state-of-the-art semi-automatic approaches for code anomaly prevention and detection. We expect these studies will guide us on the design of a recommender system intended to enhance collaborative judgment about the criticality of code anomalies.

Short Bio

Roberto Oliveira is a postgraduate student in the Informatics Department of PUC-Rio since 2013, with graduation scheduled in August 2017. He already completed the pre-requisites, the courses and he is now working on his PhD thesis proposal. Currently, he is a member of the Software Engineering Laboratory (LES). He is particularly interested in advanced techniques for modular programming, pair programming, multi-agent systems, experimental software engineering, and software product-line design. He obtained a Msc degree in Computer Science (early 2013) from the Federal University of Paraíba (UFPB) and a BSc degree in Computer Science (2003) from Paulista University (UNIP).
Towards a Process Models to Conduct Software Engineering Experiments

Liliane Sheyla da Silva Fonseca

Federal University of Pernambuco, Recife, Brazil

Several investigations were performed emphasizing the importance of using empirical studies to assess new technologies, tools, etc., thus helping to form a knowledge base and to accelerate their adoption. Processes are commonly used to guide activities/tasks towards the achievement of goals, and also help to integrate theory and practice. Although some researchers have proposed mechanisms to support empirical studies, only a few are being used. Therefore, it is important to understand the available empirical studies support mechanisms to facilitate and to improve the quality of research. The main idea of this proposal is defining and modeling a process model to conduct software engineering experiments based on experimental software engineering (ESE) best practices. This process model will contribute to gathering several support mechanisms scattered in the literature and will support researchers choosing adequate empirical technologies for evaluating their researches. This research aims to improve the quality of empirical results and to provide guidelines for evaluating research. This initiative intends to reduce the lack of scientific results in ESE. First of all, a systematic mapping study was performed to identify the main mechanisms used to support software engineering experiments. This proposal is based on the Improvement Paradigm and ESE best practices found in the literature. The process will be defined and modeled with BPMN (Business Process Modeling Notation) and assessed by empirical studies. We expect to provide a knowledge base with key elements of a process model to conduct software engineering experiments. Thus, the development of processes to conduct empirical studies can facilitate and promote further empirical research in software engineering.

1 Short Biography

Liliane Fonseca is from Brazil. She is currently a third year Ph.D. candidate in Computer Science at the Federal University of Pernambuco, Brazil. She will finish her PhD in February 2016. Her research involves empirical studies in software engineering. Her doctoral work focuses on defining and modeling process models to conduct software engineering experiments. She attends an empirical software engineering group at UFPE. Recently, her group published a systematic mapping study about support mechanisms to conduct empirical studies in software engineering at ESEM (Empirical Software Engineering and Measurement) in 2014. Liliane earned a Masters degree in Computer Science in 2012 from the Federal University of Pernambuco. Her master’s research was focused on risk management. She has a BSc in Computer Engineering from the University of Pernambuco, Brazil (2009).
Crowdsourcing: Software Industry transformation and disruption

Leticia Santos Machado

Pontificia Universidade Católica do Rio Grande do Sul – PUCRS
Av. Ipiranga, 6681
90619-900 Porto Alegre, Brasil
{leticia.machado.001@acad.pucrs.br}

Abstract. The evolution of Web 2.0 media technologies potentiated the collaboration of knowledge reflecting the phenomenon of Crowdsourcing where a large and undefined group of potential contributors participate in online way to solve problems and tasks that are globally distributed. Several knowledge areas have been using Crowdsourcing to start obtain scalability, to promote innovation, to cut off expenses, and to save time. The area of Software Engineering has recently started exploring the Crowdsourcing’s model in tasks of software development seeking collective solutions to problems, ways to accelerate the time-to-market, increase quality and also reduce costs. However, although this paradigm indicates a trend in the area of Software Engineering, fundamental concepts behind Crowdsourcing has not yet been well explored. The purpose of this research is to collect empirical evidence about the topic in order to explore the key insights that shed light on the phenomenon of Software Crowdsourcing. We also want to understand how crowdsourcing can will be applied in software development domain including possible benefits, the main concerns, the impact and the factors that block some companies to adopt it, from three perspectives: the buyers (requesters), the platforms, and the workers (crowd).

I am a Ph.D. candidate at the Computer Science School at PUCRS. My advisor is Prof. Rafael Prikladnicki and my main research area is Software Engineering. In my Ph.D. research, I am investigating the crowd-based software development. I work as an assistant professor in Computer Information Systems at the Universidade do Vale do Rio dos Sinos (UNISINOS) in Porto Alegre. I started my Ph.D. in March, 2014 and I intend to conclude in 2018.
Using social, history and technical aspects from software development to explain the occurrence of co-changes

Igor Scaliante Wiese
Phd Student from USP - University of Sao Paulo
igor.wiese@gmail.com

1 Abstract

During the software evolution, developers contribute to open-source software projects submitting source code to fix defects and add new features. Since a source code is submitted, interested parties review the set of changes and discuss potential changes. Because of this evolution, some artifacts evolve together throughout software development process. The concept of co-changes (a.k.a. logical coupling) captures the notion of some artifacts frequently changing together during software development. It is known that the co-changes, if not properly understood, can affect software quality since this type of dependency occurs in an unplanned way. Thus, understanding the reasons of occurrence of co-changes is critical for software architects, developers and manager since they can identify solutions to avoid the co-changes or treat them. Previous research used structural dependencies to explain why co-changes occurs on software development. They concluded that the interplay between structural dependencies and co-changes cannot explain the origins of co-changes. Considering software development process, previous research has indicated the relationship between the different dimensions of development with software quality (e.g. defect prediction). Similarly, co-changes have been correlated to the quality of the software. However, despite of this effort, we still cannot understand the origins of the co-changes. The novelty of this work consists in understanding the relationship between different dimensions of software development and the occurrence of the co-changes. In general, we want to investigate if the social, historical and technical software development dimensions can explain the occurrence of co-changes. We are using statistical and classification approaches to model the relation between the metrics that instantiates each dimensions and the occurrence of co-changes.

2 MiniCV

I am a PhD candidate at University of Sao Paulo, under supervision of Marco Aurlio Gerosa. My current interest is on explore the origins of co-changes, mainly on the relation between software metrics and co-changes. I have some papers published on this research topic. I started my PhD in August, 2011 and aim to finish it by the end of 2015. Currently I am also a faculty at Federal University of Technology - Paran.
On the evaluation of Distributed Collaborative Programming

Bernardo Estácio

Pontificia Universidade Católica do Rio Grande do Sul – PUCRS, Av.Ipiranga, 6681
90619-900 Porto Alegre, Brasil
bernardo.estacio@acad.pucrs.br

Abstract.
Collaborative programming is an important technique in software engineering to promote a learning environment and knowledge transfer between software developers, also improving the quality of the project. In this context, Pair Programming has been established as a practice that effectively supports that. In addition, Coding Dojo has recently emerged as a collaborative group practice that uses Pair Programming to allow everyone to participate in the software development phase. One example where these practices can benefit the software teams is Distributed Software Development (DSD). The large adoption of DSD by IT companies requires distributed teams with a common knowledge of the project to get succeed. In this context, Distributed Pair Programming and Distributed Group Programming (evidenced by Coding Dojo) have an important role to support the challenges of the distribution. Although Pair Programming and Group Programming are not new concepts in SE, few studies empirically assess these practices or proposed guidelines to support the DSD challenges. In this context, the goal of this research is identify which characteristics are needed to support pair and group programming when teams are distributed, in order to facilitate the promotion of a learning environment in distributed software projects, also improving the quality of the projects delivered. We also want to empirically evaluate the benefits of adopting these practices in software development and understand the role and the impact of tools in this process.

Bernardo Estácio is a PhD Candidate at Computer Science School at PUCRS, under the supervision of Professor Dr. Rafael Prikladnicki. He started his PhD in March 2013 and intends to conclude in 2017. His research interests are in the area of global software development and agile practices, mainly collaborative programming practices. He also has over three years of experience in projects with industry, dealing with research and software engineering. As part of his research and scholarship, Bernardo works part-time as a Scrum Master (Coach) in software development projects.
Safe Monitoring of Physical Exercises

Elthon Allex da Silva Oliveira

Abstract. Several software solutions have been developed for the area of healthcare. In this context, we highlight the field of physical exercise that has received great attention from WHO for helping decreasing risk of various diseases and to improve the quality of life of people affected by disease. However, there are some problems with existing solutions developed for monitoring of physical exercises: only physiological variables are supported, professional attention required during the exercise, and absence of customized recommendations. A method has been developed to provide secure monitoring of individuals and customized recommendations during the exercises without the presence of professional in loco. With a tool developed, the professional builds at his own clinic a model of automata that works as a reference model for the individual. This model is made with information about individual and its goals. Liveness and safety properties are checked in the model by model checking technique. Custom properties can also be checked. Another tool was developed to run on smartphones. This second tool runs the model generated by the first tool and, from information measured by sensors (environmental and body temperatures, dehydration, etc) on the environment and the individual, offers recommendations to the user. Such recommendations aim to keep the individual in a safe physiological state or get it from a physiological one in which the exercise is not providing the desired benefit. Both tools have been developed as proof of concept of the proposed method, part of work evaluation. In addition, some simulations were performed and compared with the opinion of a specialist in the field of exercise physiology who also evaluated the proposed method. Such a specialist was the gold standard.

Short Bio

Elthon received the B.S. degree in Computer Science from Federal University of Alagoas (UFAL), Maceió, Brazil, in 2004, and the M.S. degree in Informatics from Federal University of Campina Grande (UFCG), Campina Grande, Brazil, in 2006. He is professor in Campus Arapiraca, UFAL, since 2006 and is a PhD candidate in Computer Science at UFCG since 2010. Elthon intends to complete his course by the end of the current year.
Exploring Architecture Blueprints for Prioritizing Critical Code Anomalies

Everton Guimarães

Abstract. 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. A code anomaly is a recurring implementation structure that possibly indicates deeper design problems. Code anomaly is considered critical when it is related to a structural problem in the software architecture. Its criticality stems from its negative influence on non-functional requirements. The presence of critical code anomalies hinders software maintainability, i.e. wide refactoring might be required to remove an architectural problem. Many approaches are proposed for detecting code anomalies in software systems, but none of them efficiently support the prioritization of critical code anomalies. Our work investigates how the prioritization of 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 defined as informal design models used only to capture and communicate key design decisions. Even though blueprints are often incomplete and inconsistent with respect to the underlying implementation, their use can contribute to the process of prioritizing critical code anomalies. Nevertheless, aiming to address our research goals, a set of empirical studies has been performed. We also proposed and evaluated a set of heuristics to support developers when prioritizing code anomalies in 3 industry-strength systems. The results showed an average accuracy of more than 60% when prioritizing code anomalies associated with architectural problems.

Keywords. Code Anomalies, Blueprints, Software Architecture, Heuristics, Empirical Evaluation

1 Short Bio

Everton is PhD Student in Science – Informatics at Pontifical Catholic University of Rio de Janeiro (PUC-Rio) with completion scheduled to September 2014. He received his M.S. in Computing and Systems from the Federal University of Rio Grande do Norte in 2010. Everton’s research areas include software evolution, software modu-
larity, software metrics, model-driven development, software anomalies, software product lines and empirical studies. Moreover, Everton has participated in the organization of premier conferences in the field of software engineering, such as *International Conference on Aspect-Oriented Software Development* - which has been officially renamed to MODULARITY (2010) and *International Conference on Model Driven Engineering Languages and Systems* (2013). Currently, he is member of the OPUS Research Group under the supervision of Prof. Alessandro Garcia. He has also been involved in the organization of many workshops and events promoted by the OPUS Research Group with the participation of several national and international research collaborators. Everton has been involved in many research projects in the last 4 years as part of his PhD research. Finally, he has recently been involved as Research Scholar in a collaboration program between PUC-Rio (Brazil) and Drexel University (USA) were he developer part of his PhD research.
Short Bio – Natasha Malveira Costa Valentim

Natasha Malveira Costa Valentim has a degree in Computer Science from Federal University of Amazonas (Brazil-UFAM), where she also obtained her M.Sc. degree in Computer Science. Natasha is part of the research group "Usability and Software Engineering" (USES), which researches and develops technologies that evaluate the usability of models used in the early stages of software development.

Natasha participates in research projects that have focused on improving the quality of software as projects entitled "Improving Quality in Software Development in State of Amazonas by Means of Integrating Usability in the Development Process" and "Innovative Strategies for Improving Quality in Software Development in State of Amazonas". Natasha has co-authored 6 papers. The complete list of her papers can be found on http://lattes.cnpq.br/1374747002879287.

The researcher is studying for a Doctorate in Computer Science since March 2013 in UFAM and is expected to end in March 2017. Natasha received an award for her work in the area of software quality in Brazilian Symposium on Software Quality (SBQS 2014).
Improving the usability through the creation of models in early stages of the software process

Abstract. Recent researches aimed at ensuring a high degree of usability in the early stages of the development process of applications, called “Early Usability”. Part of the proposed techniques is intended to ensuring usability through the inspection of models used during the design of the applications, aiming to a higher user satisfaction and reducing development time. However, these approaches have been independently developed for specific models. In this way, to ensure the usability of the final software based on the models, during their design/creation, allow the identification of a larger number of usability problems in the early stages of the development process. In this proposal, we intend to propose new technologies (methods, techniques, tools and processes) for the control of usability with emphasis in the early stages of development. The final purpose is that the proposed set of technologies can be employed by the software engineers during the development of new applications. With this, we hope to contribute to improving the quality of developed software and the decreasing cost of correcting usability problems before the source code is written.

Keywords. Early Usability, Design Models, Usability Design
An Approach for History-Sensitive Reliability Verification for Software Product Lines

André Luiz Peron Martins Lanna

University of Brasília
E-Mail: andrelanna@unb.br

Abstract. Software Product Line Engineering (SPLE) is a development approach which aims at developing a set of applications for a specific domain or market segment in a systematic manner by reusing a set of reusable software assets. Current model checking techniques have been used to verify properties of Software Product Lines (SPL), but are not able to identify the impact on a verification model each time a change occurs in the SPL. They are called history-insensitive because for each change in a SPL, a new verification model has to be built from scratch in order to maintain the consistency between the design and verification models of the SPL. However, building a new verification model for each change may not scale. A possible solution is building the verification models in an incremental and compositional manner. To evaluate the SPL reliability, its behavior is modelled by Discrete Time Markov Chains (DTMC). My research aims at developing an approach to tame SPL reliability verification effort by reusing verification models previously calculated. Each time a SPL change, the DTMCs fragments impacted are identified and adapted accordingly in order to reflect the SPL changes. Such DTMC adaptations occur by composing new DTMC fragments in place of the changed DTMC fragments. This approach can be considered history-sensitive because it can identify the unchanged DTMCs fragments of a SPL and reuse them. With DTMC fragments reuse, it is expected that such approach will allow the scalable reliability verification. Two kinds of evaluation are considered for this research. Formal methods will be used to ensure the DTMC properties are permanently satisfied. To evaluate the proposal’s feasibility and scalability, empirical methods will be accomplished on well-established SPLs.

Bio: André Lanna has a bachelor course degree on Computer Science and masters degree on Electrical Engineering, both at Pontifical Catholic University of Minas Gerais. His topics of interest are Software Product Lines and software verification methods (specifically Model Checking). Since August, 2012 he is a Ph.D. student on Computer Science at University of Brasília and its final is expected for July, 2016.
Strategies for Prioritizing Test Cases Generated Through Model-Based Testing Approaches

João Felipe Silva Ouriques

Software Practices Laboratory (SPLab), UFCG, Campina Grande, PB, Brazil
jfelicpe@copin.ufcg.edu.br

Software testing is very expensive and time consuming, especially for large software. In order to deal with the cost of the testing activity, researchers proposed the Model-Based Testing (MBT). In MBT, test cases are generated automatically from the models, reducing time and cost; however a common drawback of the generation is a huge generated test suite. Due the restricted budget, executing the entire test suite is hindered. Therefore, test cases with higher priority should be executed beforehand. Many Test Case Prioritization (TCP) techniques have been proposed in the regression testing context and the goal is to find failures as soon as possible, when some modification is performed, but this is out of the scope of this work. This focuses on a specific issue: test case prioritization in MBT context when historical information is not available.

Before proposing a new technique, we have investigated TCP in MBT context, and what factors may affect the performance of the techniques. So far, the influence of the model structure and the characteristics of the test cases that fail were evaluated through experiments. The results suggested that the model structure does not affect significantly the performance of techniques, but these are sensitive to different characteristics of the test cases that fail.

A worthy information in this context would be an expert who knows the crucial parts of the software, i.e., what parts of the model are more error-prone. A hint of an expert could guide the entire/partial prioritization process. Considering our former results, we have proposed the first version of the technique that considers hints and distance in order to prioritize. Evaluation and tuning of the technique are going on, but preliminary evaluation reveals promising results.

Bio

João Felipe is Ph.D. student in Computer Science at Federal University of Campina Grande, Brazil, since 2012 and it is estimated to be concluded in December 2015. He received B.E. and M.Sc. degrees in Computer Science from the same institution in 2009 and 2011 respectively. His research interests in Software Engineering include, Software Testing, Model-Based Testing and Test Case Prioritization.
Towards Effort Estimation in Software Testing Projects

Thiago Silva-de-Souza1

1 Programa de Engenharia de Sistemas e Computação (PESC), Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia (COPPE), Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro-RJ, Brazil
thiagosouza@cos.ufrj.br

Abstract. Currently software testing has become an industry of its own over the years, with the emergence of independent testing services companies establishing testing services as a business unit. Effort estimation is a critical activity in software testing and, therefore, some models have been proposed in the technical literature to support such activity. The goal of this PhD work is establish an approach for software testing projects effort estimation. We performed a secondary study that has identified software testing effort influence factors and estimation models using these factors. The identified models and factors cannot be generally applied due projects’ variability. Besides, evidence regarding the lack of consensus about software testing and testing effort makes the chosen of any estimation model risky and error prone. In the coming months we will conduct a survey in the industry to corroborate the results of the secondary study. Then we will try to define an estimation model using the influence factors previously identified.

Keywords: software testing, effort estimation, cost drivers.

1 Short Bio

I am a PhD student in Software Engineering at COPPE/UFRJ (Brazil) focusing on Software Quality, especially Software Testing. My advisor is Professor Guilherme Horta Travassos. My PhD thesis aims to investigate means to establish an approach for software testing projects effort estimation.

The main activities of my PhD work are organized as the following schedule:

- **September/2012**: PhD start
- **September/2012 – December/2013**: Taking doctoral courses
- **December/2013 – July/2014**: Systematic Literature Review
- **August/2014 – December/2014**: Survey in industry
- **January/2015 – July/2015**: Estimation model development
- **August/2015**: PhD Qualifying
- **August/2015 – December/2015**: Estimation model evaluation (experiments)
- **January/2016 – July/2016**: Thesis writing
• September 2016: Thesis defense (PhD end)

In my Master’s dissertation I proposed a specification-based approach for RESTful Web Services test case generation. This dissertation was advised by professors Eber Schmitz (UFRJ, Brazil) and Alexandre Luis Correa (UNIRIO, Brazil). We published three papers related to this research in Software Engineering conferences.

In addition, I have been worked with Software Testing in the industry and academia for some years. I am an employee of Serviço Federal de Processamento de Dados (SERPRO) since 2005. SERPRO is one of the major public IT companies of Latin America, responsible for developing the systems of many Brazilian governmental agencies. In addition, I teach undergraduate courses on Software Engineering at the UNIGRANRIO University since 2004.