Electronic Communications of the EASST
Volume 59 (2013)

Special Issue of the First Workshop on Patterns Promotion and Anti-patterns Prevention (PPAP 2013)

Preface

4 Pages

Guest Editors: Aminata Sabané, Wei Wu
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Preface

On the 5th of March, 2013, the first workshop on Patterns Promotion and Anti-patterns Prevention (PPAP 2013) took place in Genova, Italy. PPAP 2013 was co-located with the 17th European Conference on Software Maintenance and Reengineering (CSMR'2013), the premier European conference on the theory and practice of maintenance, reengineering and evolution of software systems.

With the aim of promoting the application of patterns and prevent the spread of anti-patterns, the first objective of PPAP is to build a bridge between the different families of patterns and anti-patterns in software engineering. Relevant topics include but are not limited to:

- definition and detection;
- impact on software quality, testing;
- application and refactoring;
- empirical studies;
- evolution;

of patterns and anti-patterns in software engineering.

For this purpose, the first PPAP workshop intended to provide a place to discuss possible interactions between patterns and anti-patterns across different domains and how they could benefit from one another. In particular, researchers working on patterns and anti-patterns in different domains in software engineering could share their experiences as well as compare their results. In addition, we believe that emerging patterns and anti-patterns could benefit from the past lessons and successes of their predecessors.

PPAP 2013 attracted 30 participants and 5 papers were presented. The titles of the accepted papers were:

- "SQL Pattern Organization and Presentation", by Huda Al-Shuaily and Karen Renaud;
- "PatEvol - A Pattern Language for Evolution in Component-Based Software Architectures", by Aakash Ahmad, Pooyan Jamshid, Claus Pahl, and Fawad Khaliq;
- "An Approach to Formalise Security Patterns", by Luis Sergio da Silva Junior, Yann-Gaël Guéhéneuc, and John Mullins;
- "Analysing Anti-patterns Static Dependencies", by Fehmi Jaafar, Yann-Gaël Guéhéneuc, and Sylvie Hamel;
- "Influence of numerical thresholds on model-based detection and refactoring of performance antipatterns", by Davide Arcelli, Vittorio Cortellessa, and Catia Trubiani.

PPAP 2013 was honoured by two keynote speakers. The first keynote was given by Rudolf Ferenc, assistant professor at the Department of Software Engineering, University of Szeged, Hungary. Rudolf is a well-known pattern-related researcher. He leads several R&D projects

which are related to quality assessment, the improvement and architecture reconstruction of software systems of major banks, and software development companies in Hungary. His talk was about the effect of design patterns on software maintainability:

Title: Myth or Reality? The Effect of Design Pattern Usage on Software Maintainability

Abstract: Although the belief of utilizing design patterns to create better quality software is fairly widespread, there are relatively few research papers objectively indicating that their usage is indeed beneficial. In this talk we will try to reveal the connection between design patterns and software maintainability. For this sake, we have performed two experiments.

First, we have analyzed more than 300 revisions of JHotDraw, a Java GUI framework whose design heavily relies on some well-known design patterns. We used our probabilistic software quality model for estimating maintainability and we parsed the javadoc annotations of the source code for gathering pattern instances. The Pearson correlation of the design pattern line density and the maintainability values revealed an interesting relationship.

Second, we have measured the maintainability of the software systems in DBP – the Design Pattern detection tools Benchmark platform – and calculated the correlations of the maintainability values with the numbers of the detected design pattern usages. We checked the results of four tools: DPD tool, Web Of Patterns, P-MARt, and MARPLE-DPD. Again, the outcomes are noteworthy. But after considering only the hand-validated true pattern instances, the results are becoming really exciting.

The second keynote was given by Radu Marinescu, associate professor at the Politehnica University of Timisoara, co-founder and head (since 2002) of the LOOSE Research Group and co-founder of Intooitus, a young spin-off company creating disruptive quality assessment tools. Radu was one of the precursor in the automated detection of anti-patterns. He shared his experience about design flaws detection:

Title: Assessing technical debt by identifying design flaws in software systems

Abstract: Tough time-to-market constraints and unanticipated integration or evolution issues lead to design tradeoffs that usually cause flaws in the structure of a software system. Thus, maintenance costs grow significantly. The impact of these design decisions, which provide short-term benefits at the expense of the system's design integrity, is usually referred to as technical debt. In this paper, I propose a novel framework for assessing technical debt using a technique for detecting design flaws, i.e., specific violations of well-established design principles and rules. To make the framework comprehensive and balanced, it is built on top of a set of metrics-based detection rules for well-known design flaws that cover all of the major aspects of design such as coupling, complexity, and encapsulation. I demonstrate the effectiveness of the framework by assessing the evolution of technical debt symptoms over a total of 63 releases of two popular Eclipse® projects. The case study shows how the framework can detect debt symptoms and past refactoring actions. The experiment also reveals that in the absence of such a framework, restructuring actions are not always coherent and systematic, not even when performed by very experienced developers.

Besides paper presentations and keynotes, the workshop included two working sessions where participants were invited to discuss future research on patterns and anti-patterns. In the following, we summarize the main results of the discussions:

- We researchers must observe more; new patterns/anti-patterns must be defined based on the good/poor practices of practitioners rather than only to promote/prevent existing patterns/anti-patterns.
- Context-aware patterns/anti-patterns are needed, because a solution may not always be good/poor in different domains.
- Formalizing patterns/anti-patterns definition has the benefit of facilitating the automation of operations on them, such as detection and application.
- More empirical studies are needed showing (1) causality of the benefit of patterns and of the negative impacts of anti-patterns on different quality aspect of software and (2) the possible contagiousness of anti-patterns on other program entities of source code are required. Such studies would better help the practitioners and managers make more rational decisions regarding patterns and anti-patterns.
- Further research was suggested on how to combine and apply patterns on real systems including support tools for such implementation.
- Finally, participants highlighted that patterns are learned better through struggling than mentoring, thus we must consider the learning strategies when promoting the benefit of patterns.

Three of the five accepted papers are extended for this special issue. We would like to thank the program committee for their detailed and constructive reviews: Giuliano Antoniol, Yann-Gaël Guéhéneuc, Foutse Khomh, Naouel Moha, Leon Moonen, Daniele Romano, Lionel Seinturier, Nikolaos Tsantalis, Aiko Yamashita, and Andy Zaidman.

We also thank all the authors and participants for their contribution and participation.

PPAP Organizing Committee:
Surafel Lemma Abebe, Venera Arnaoudova, Laleh Eshkevari, Aminata Sabané, Wei Wu