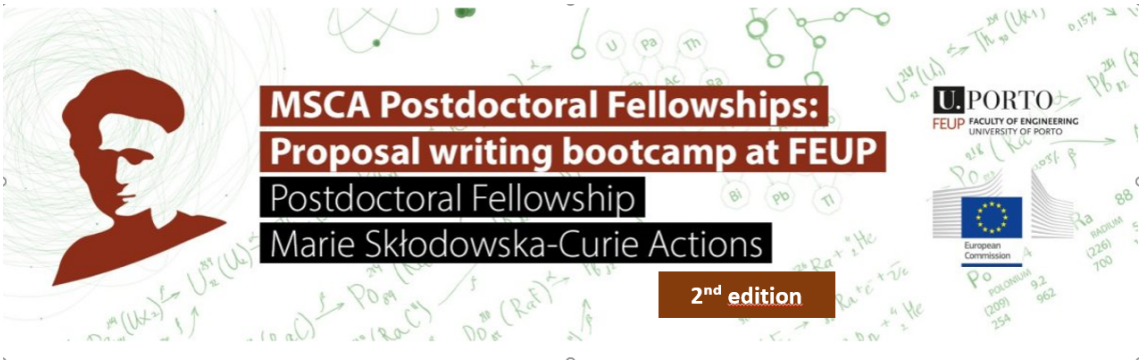
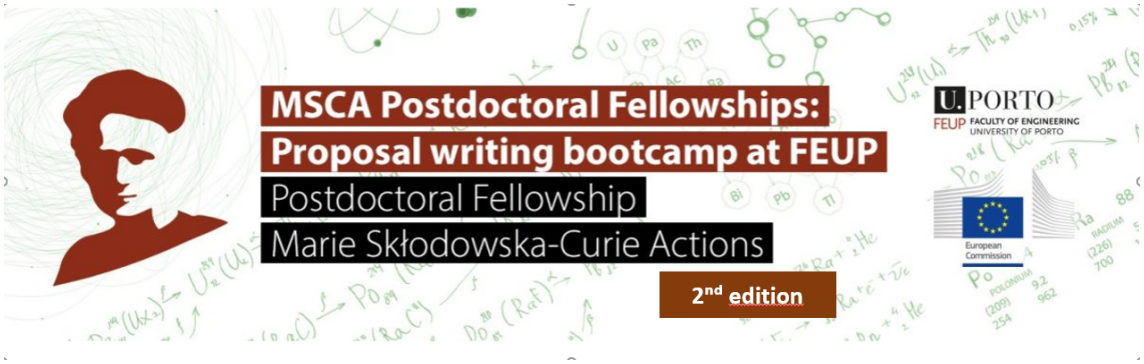


SUPERVISOR INFORMATION	
First and Last name	José Campos
URL of supervisor webpage	https://jose.github.io
Department	Department of Informatics Engineering
Field(s) of research	Software Engineering, Software Testing, Search-based Software Engineering, Empirical Software Engineering, Mining Software Repositories
PROJECT PROPOSAL	
Title (optional)	Automatic Minimization of Bug-fixing Software Patches
Brief project description	

Open-source code repositories provide a rich source of data for empirical software engineering research. For example, research on software testing, fault-localization, and automated program repair often employs bug-fix patches from open-source repositories for evaluation. Such analyses rely on the assumption that bug-fix patches only contain the exact changes required to fix the bug, or in other words, do not contain changes such as refactoring or stylistic changes (which are related to the bug that was fixed). However, it is known that developers often create bug-fix patches that mix changes related and not related to the bug-fix (this is often known as tangled code changes [1]).

To ease the evaluation of empirical software engineering research, researchers have mined platforms such as GitHub, gathered thousands of bug-fix patches from thousands of open-source repositories, and proposed several datasets of bug-fix patches. For example, SIR and ManyBugs are two examples of datasets of bug-fixes in C programs, BugsInPy and ExcePy in Python, BugsJS in JavaScript, QBugs in quantum computing, and Defects4J, BugSwarm, BEARS, Bugs.jar, and QuixBugs in Java. Despite the diversity of datasets, only the well-known [Defects4J](https://github.com/rjust/defects4j) dataset [2] provides minimal bug-fix patches (854 at this moment). The Defects4J dataset's team has spent hundreds of hours manually minimizing each bug-fix patch so that others could use those minimal patches in their empirical studies. However, the minimization procedure used, which is described in [here](https://github.com/rjust/defects4j/blob/master/framework/bug-mining/Patch-Minimization-Guide.md), is extremely time-consuming and prone to errors.

Thus, this project aims to investigate and develop novel approaches to minimize non-minimal bug-fix patches in any programming language. The novel approaches would reduce the time and potential errors associated with manual minimization of bug-fix patches while also



improving the quality of data available for empirical software engineering studies. Although the first prototype would be developed for Java and integrated into the Defects4J dataset, this project aims to develop tools to other programming languages and datasets.

[1] Kim Herzig, Andreas Zeller. "The impact of tangled code changes". In: Working Conference on Mining Software Repositories (MSR), 2013. https://ieeexplore.ieee.org/abstract/document/6624018

[2] René Just, Darioush Jalali, and Michael D. Ernst. "Defects4J: A Database of Existing Faults to Enable Controlled Testing Studies for Java Programs". In: International Symposium on Software Testing and Analysis (ISSTA), 2014. https://dl.acm.org/doi/abs/10.1145/2610384.2628055