**GUITAR - A GUI Testing Framework**

Filip Anda Roxana

Dubău Horea Filip

Deac Iulian Teodor

DOI: 10.1007/s10515-013-0128-9

GUITAR is designed to facilitate and automate GUI testing. GUI testing can be time-consuming and error-prone when done manually. This is because GUI-based applications can have many elements, each with many possible interactions. Classic manual testing methods can be ineffective in such cases, leading to insufficient testing coverage and potentially omitted defects. The motivation behind GUITAR is to provide a reliable and efficient way to test GUI-based applications and to accommodate the constant influx of new models. The approach used by GUITAR is model-based testing, which implicates building a model of the GUI's behavior and generating test cases automatically from this model. This approach provides several benefits, including improved test coverage, reduced testing time, and more significant reliability. GUITAR's model-based method allows testers to reuse test cases across different applications and platforms improving efficiency. GUITAR’s approach to testing oracles is similar to the Monkey approach, in that failures of a certain class (e.g., crashes, permissions errors, timeouts) can be detected by a class-specific Test Monitor. However, Test Monitor implementations, because they are plugins to the framework, are completely customizable.

The GUITAR framework aims to help software developers and testers enhance the quality and reliability of their applications by identifying defects and issues in the GUI. It also aims to reduce the duration and effort needed for manual testing, as well as the costs associated with software testing. The plugin-based approach also allows GUITAR to be extended to new platforms as they become available and allows developers to customize and extend GUITAR at the model, event, and widget levels of abstraction, as we illustrate throughout this paper. GUITAR uses an event-flow model to capture GUI events, which allows it to generate test cases that cover a large portion of the GUI behavior. This is in contrast to other frameworks that use a state-based model, which can result in a large number of redundant test cases. GUITAR is designed to be scalable, meaning it can handle large and complex GUIs. This is achieved through various optimizations, such as using event compression, clustering, and parallelization.

It is recommended that future research compare GUITAR with industry-standard frameworks, with priority on aspects of the framework that are particularly beneficial to quality assurance specialists. These aspects should include the framework's effectiveness in identifying weaknesses, its efficiency in executing feasible numbers of test cases within a reasonable turnaround time, and its scalability in real-world applications.