**Summary document**

Paper title: Using Static Analysis to Find Bugs

DOI: [10.1109/MS.2008.130](https://ieeexplore.ieee.org/document/4602670)

Team members: Buiga Andreea-Diana, Ceontea Vlad-Călin, Chirea Liviu

Although essential, software quality is frequently flawed in reality. There are countless ways of improving the quality and one of them include static analysis tools. Such tools search for breaches of reasonable or advised programming practice rather than attempting to demonstrate that the code complies with its specification. One such tool is FindBugs, which looks for coding defects. The FindBugs initiative started as a small experiment, and then took off, garnering more than 500,000 downloads globally. The initial discovery was that some Java programs contained glaring errors that could be found using relatively simple analytic methods.

FindBugs began with the discovery that certain Java programs featured glaring errors that could be found using simple analysis tools. Early tests revealed that similar errors were present in even "production quality" software and that even seasoned engineers made them. Instead of using a pattern language like other bug detection tools, FindBugs uses various techniques written in Java, including visitor pattern and data-flow analysis. The tool groups bug patterns into categories and assign each pattern a priority level. It is important how new bug detectors are developed in the project, starting with real bugs and developing the simplest possible technique that effectively finds such bugs. Moreover, FindBugs makes use of more advanced analytic techniques designed to precisely pinpoint certain problems, such as the dereferencing of null pointers, that are prevalent enough to merit their development.

The tool was validated by going from finding a particular instance of a bug to implementing a detector that can effectively find instances of it within hours. The tool was automated in the first stage to run all new Java code, and the generated warnings were recorded. However, the warnings were not effectively used by developers due to limitations in the database and Web interface. Due to restrictions in the database and Web interface, developers did not use the warnings very efficiently. During the second phase, two team members split their time equally between assessing alerts and notifying Google of serious flaws. On the basis of developer comments and actual findings, the evaluation prioritized bug patterns. The third phase involved integrating analytical feedback into the development workflow using Google's code-review policy and tools. FindBugs was validated through its ability to identify significant defects in Google's bug-tracking system.