CSD 380

Discussion 11

Dynamic Analysis is a software testing technique used to evaluate a program's behavior during its execution. Unlike static analysis, which examines code without running it, dynamic analysis requires the application to be executed, allowing testers to observe how the software behaves in real-world or simulated environments. This approach is commonly used to detect issues such as runtime errors, memory leaks, performance bottlenecks, and security vulnerabilities. Dynamic analysis requires the program to be run, either in a test environment or in production, to monitor its behavior under various conditions. It focuses on the runtime interactions between the program, the operating system, and external resources. This method observes the program as it processes input, handles user interactions, and manages system resources. By completing dynamic analysis, there are a couple of things that we can detect. Some of those being performance issues, memory problems, security vulnerabilities, and functional defects. Basically see if the software is utilizing too many resources, have slow execution times, bottle necks and even memory usage.

Some advantages towards completing dynamic analysis as opposed to static analysis is the ability to detect issues during actual runtime. IT provides insight into how the software interacts with the environment and system. By getting this information early, we would be able to address them before the issues go out to the public. To which overall provides a better user experience and uncovers runtime defects. Dynamic analysis is an essential part of ensuring software quality and security. It complements static analysis by focusing on runtime behavior, making it critical for detecting issues that only manifest during execution. Properly incorporating dynamic analysis into the development lifecycle helps create robust, secure, and efficient applications.