Corey Gaspar

9/10/2025

**2-2 Journal: Dynamic and Static Testing**

**Static Testing**

Static testing, also known as verification testing or non-executing testing, is a method of checking software for errors without running the code. The main goal of static testing is to find problems early in development, which will save time, reduce costs, and improve the quality of the software. Static testing involves reviewing code and documents such as project requirements, design specifications, test cases, and prototypes. Static testing can be done manually through reviews and inspections or automatically using tools like Checkstyle, SonarQube, Lint, and Sourcemeter, as mentioned in the article from GeeksForGeeks. Although it can’t detect runtime errors, static testing helps ensure that the development team is following coding standards, provides immediate feedback, and prevents defects from being developed into the code.

**Dynamic Testing**

Dynamic testing, also known as execution or functional testing, is a method of testing software for errors by running the code and analyzing its behavior during execution. Its main purpose is to find defects within the code, verify that the system is working as planned, and ensure that the software meets requirements under different conditions. Dynamic testing includes several strategies such as unit testing, which tests individual components; integration testing, which evaluates how components work together; system testing, which tests the system as a whole; and acceptance testing, which ensures that the final product meets the needs of the end user. Dynamic testing helps identify functional issues, performance issues, and potential security risks that can’t be detected through static testing.

**Static vs. Dynamic Testing**

Static and dynamic testing are complementary to each other and are both used to ensure software quality. Static testing focuses on analyzing code and documents without having to run the code. This allows defects to be detected early and helps reduce development costs. On the other hand, dynamic testing involves running the code to observe its behavior and ensure that it functions as planned. Dynamic testing involves exposing the code to different conditions or user input to ensure that it performs well and functions correctly. Although static testing is much faster than dynamic testing and can catch errors in design or documentation, dynamic testing is important for identifying issues within the code during runtime and ensuring that the software functions properly.

**Static and Dynamic Testing Together**

Using both static and dynamic testing in a project is important because they both catch different types of errors. Static testing identifies issues in the code early, preventing these errors from being built into the program. Once those errors are corrected, dynamic testing ensures that no new errors were introduced during development and that the software works as intended.

**Work Cited**

GeeksforGeeks. (2025, July 11). Static testing - software testing. <https://www.geeksforgeeks.org/software-engineering/software-testing-static-testing/#what-is-static-testing>

GeeksforGeeks. (2025, July 11). Dynamic testing - software testing. <https://www.geeksforgeeks.org/software-engineering/software-testing-dynamic-testing/>