CS-320 Software Test, Automation QA

Huan Ai

9/7/2025

What occurs during the testing stage of the SDLC?

The testing stage is basically where we check the software for problems before it gets released to users. It’s not just about finding crashes. Testers try to break the software on purpose by doing things a normal user might do, like entering weird data or clicking buttons in the wrong order. They run tests to see if the software behaves the way it was designed to. The main goal is to find any bugs or mistakes so developers can fix them. As Sommerville (2016) points out, testing is meant to uncover defects in the software, ensuring it meets its requirements. It’s the final quality check to make sure everything works correctly before it goes out the door.

Why is the testing stage vital to a successful SDLC?

Testing is very important because it saves a lot of time, money, and protects the company's reputation. Finding and fixing a bug early is much cheaper and easier than trying to fix it after the software is already released to customers (Sommerville, 2016). If you skip testing and release a buggy product, users will have a bad experience and might not trust your company again. It also makes sure the software actually does what it was designed to do, which is the whole point of the project. As Ammann and Offutt (2016) explain, testing provides critical information about the quality of the software, which is necessary for making decisions about its release. Without testing, you’re just hoping everything works, which is a huge risk.

Are there any exceptions where testing would occur earlier or later?

Yes, definitely. The typical SDLC has testing near the end, but some modern approaches changed this. For example, testing can happen much earlier in models like Agile. In Agile, testing isn’t just one stage at the end; it’s something that happens continuously throughout the entire project. As soon as a small piece of code is written, it gets tested. It helps find issues right away instead of letting them pile up until the very end.

Testing can also be pushed a little later in some situations, though this is riskier. For example, a company might release a very basic "minimum viable product" (MVP) to get early user feedback quickly, planning to test more thoroughly after they see how people use it. This isn't ideal, but it shows that the timing of testing can be flexible based on the project's goals.

**Reference**

Ammann, P., & Offutt, J. (2016). Introduction to software testing (2nd ed.). Cambridge University Press. https://beckassets.blob.core.windows.net/product/preamble/17100274/9781107172012\_intro\_001.pdf

Sommerville, I. (2016). Software engineering (10th ed.). Pearson.

https://dn790001.ca.archive.org/0/items/bme-vik-konyvek/Software%20Engineering%20-%20Ian%20Sommerville.pdf