Matthew Weigel

Professor Hayes

CSCI 315-01

February 23, 2022

How Much Testing is Enough?

In addition to writing code for software programs, one of the most important jobs of software engineers is to test the code that they have written in order to find potential bugs and errors. Failing to test sufficiently can lead to a subpar product being released, which will undoubtedly be unreliable and prone to errors. At the same time, software testing costs money which leads to the ethical dilemma of how much testing is enough while maintaining a profitable product. To find an answer, it's helpful to study an incident where incomplete testing led to catastrophe, while also considering possible solutions using codes of ethics and the Bible. I will also give my overall opinion on the situation including what I believe to be the moral obligations of computer professionals.

In the software industry, programmers oftentimes have their hands full with development and are unable to do thorough testing themselves. Therefore, companies usually hire teams of testers to thoroughly test the software before shipping it to the customer. However, thoroughly testing a product costs money and some companies choose to forgo complete testing to keep costs down. An example of where incomplete testing caused mishap is the Therac-25 accidents of the 1980s in which multiple patients unknowingly underwent massive overdoses of radiation from the radiation therapy machine Therac-25. Later investigations found that the accidents were the result of software errors. According to Leveson and Turner, "It appear[ed] that unit and software testing was minimal, with most effort directed at the Integrated system test." This

massive oversight led to the deaths of four people and injuries of two others and certainly raises the question of how much testing is enough and what the moral obligations and standards for software engineers should be.

The answer to how much testing is necessary I think depends greatly on the product. Anytime a product carries the risk of death or injury as a result of faulty design, there should be no limit to the amount of testing done until it is deemed safe. Despite the cost, you cannot put a price on human life. The only alternative where a cheaper yet less reliable product can be used should be if the patient has given informed consent.

Some people have proposed that software engineers should be required to undergo a certification process, like engineers in other fields since they are responsible for creating much of the technology today, including devices like radiation therapy machines. In response to that, I don't believe software engineers should need certification unless they are involved in creating advanced machines that cannot have room for error such as medical machines or aircraft.

Regarding the moral obligations for software engineers, a "professional is obligated to perform assigned tasks competently, according to professional standards…including not only attention to technical excellence but also concern for the social effects of computers on operators, users, and the public" (Loui and Miller). The ACM Code of Ethics also echoes this sentiment in section 2.1, saying, "A computing professional should strive to achieve high quality in both the processes and products of professional work". Even the greatest code of ethics, God's Word, the Bible gives us 2 Timothy 2:15, which says, "Do your best to present yourself to God as one approved, a worker who has no need to be ashamed…" It is wrong to sacrifice our best work in return for profits, especially if it harms others. Instead as professionals we should always strive to deliver the highest standard, which is what God would want.

Works Cited

"ACM Code of Ethics and Professional Conduct." *Acm.org*, www.acm.org/code-of-ethics.

Leveson, N.G., and C.S. Turner. "An Investigation of the Therac-25 Accidents." *Computer*, vol. 26, no. 7, 1993, pp. 18–41., doi:10.1109/mc.1993.274940.

Loui, Michael C., and Keith W. Miller. "Ethics and Professional Responsibility in Computing." *Wiley Encyclopedia of Computer Science and Engineering*, 2008, doi:10.1002/9780470050118.ecse909.