Jung Yoon (jey283)
Dr. Vallath NandaKumar
CS 349
12 March 2015

Innovation: Automatable but not Automatic

Set to graduate in a month with a BS in Computer Science, a BS in Electrical Engineering, and a BSA in Mathematics, I see myself still continuing to learn in the mobile applications industry 25 years from now into the future, equipped with the ability to see problems from different perspectives. It's difficult to know how mobile applications will change in the span of a quarter of a decade but hopefully by that time I won't be debugging code as much as I will be able to help foster innovation, thanks in part to the automation of programming that I know will happen. It is my firm belief that gradual automation will help mobile application development progress faster but that, in turn, will provide me with job security because robots will continue to lack the creative and critical thinking required to adapt to the quickly changing technologies around them.

As stated previously, I believe that automation in the software industry will happen gradually and it may take more than 25 years for it to become an accepted norm; we are far too consumed with the idea of replacing fast food workers and cashiers with robots to consider automation of programming in the short-term future (O'Toole 1). When everything is said and done, however, automation in our careers will allow for faster development; we won't have to write trivially tedious, repetitive code that we have already mastered. This guarantees us a level of job security because the exponential technological growth as is predicted by Moore's Law means that there will always be a need for adaptive, skilled developers (Moore's Law 1). The fortunate truth is that human developers make up for what robots lack - we can "come up with ideas, adjust to new realities and problem solve" utilizing constantly changing technologies (Schawbel 1). Even today, it would arguably take more work to update automatization software than to program itself without automatization at the rate that Apple makes software updates.

On the other hand, however, this inevitably means that the less skilled programmers will lose their jobs because the better, faster programmers will have more time, more mental energy

to focus on forging advancements. We must humbly remember that “If you don't keep learning, keep reading, keep improving your skills eventually that nasty steamroller behind you will flatten you permanently. Then your career is likely over.” (The Codist 1) After all, lawyers have to keep up-to-date with current laws, doctors have to study new approaches and cases, and we software developers have to quickly adapt to new technologies and master them. Specifically for mobile applications developers, there will be more competition to fill job less positions and less students who are inclined to take the risk. If anything, I predict there will be a higher demand than ever for talented automation engineers, but not so much for mobile developers.

We must also consider that “if we use automation to simply replace hard work, and therefore prevent you from fully mastering various levels of skills,...software developers [are likely to become less capable]” (Thibodeau 1). Some of today’s generation of computer scientists do not know the logic nor the machine language behind if/else statements and 25 years from now they might not have to know how sorting algorithms work. I fear that we will become more lazy as a whole and we will start to rely on automation to do much of our work for us, similar to how Java’s built-in methods arguably blinds programmers sometimes. As automation starts to build a standing in the software development industry, we must take it in ourselves to upload the integrity of code.

Bill Gates once said that “Technology over time will reduce demand for jobs, particularly at the lower end of skill set” (Bort 1). While it’s a frightening prospect to know that our skills could potentially become useless to society, we as computer scientists, as innovators must realize that we are the one profession that could stand the test of time. We are the creators, the people behind automation in the first place. Indeed, it is foolish to think that we could bypass job loss altogether but we must always humbly remember that innovation can not thrive without skilled, adaptive software developers. In order to thrive, we must strive to continue learning, to continue to adapt to changing technologies, and to not forget the fundamentals, even when automation begins to blind us.

Sources:

N.p. "Moore's Law." *Moores Law*. Memesbridge, Inc., n.d. Web. 15 Apr. 2015. <http://www.mooreslaw.org/>.

N.p. "The Programming Steamroller Waits For No One." *The Codist Posts*. Andrew Wulf, 1 June 2013. Web. 13 Apr. 2015. <http://thecodist.com/article/the_programming_steamroller_waits_for_no_one>.

Beach, Justin. "Can a Robot Do Your "Creative" Job?" *The Huffington Post*. The Huffington Post, 30 Sept. 2013. Web. 15 Apr. 2015. <http://www.huffingtonpost.ca/justin-beach/robot-jobs_b_4010378.html>.

Bort, Julie. "Bill Gates: People Don't Realize How Many Jobs Will Soon Be Replaced By Software Bots." *Business Insider*. Business Insider, Inc, 13 Mar. 2014. Web. 12 Apr. 2015. <http://www.businessinsider.com/bill-gates-bots-are-taking-away-jobs-2014-3#ixzz3XMX9D8sj>.

O'Toole, James. "Robots Will Replace Fast-food Workers." *CNNMoney*. Cable News Network, 4 Sept. 2014. Web. 15 Apr. 2015. <http://money.cnn.com/2014/05/22/technology/innovation/fast-food-robot/>.

Schawbel, Dan. "4 Abilities You Need to Avoid Being Replaced by a Robot." *The Fast Track*. Intuit Inc, 20 Feb. 2013. Web. 11 Apr. 2015. <http://quickbase.intuit.com/blog/2013/02/20/how-to-not-be-replaced-by-a-robot/>.

Thibodeau, Patrick. "How Automation Could Take Your Skills -- and Your Job." *Computerworld*. Computerworld, Inc., 10 Nov. 2014. Web. 15 Apr. 2015. <http://www.computerworld.com/article/2845383/how-automation-could-take-your-skills-and-your-job.html>.