Research Interest Narrative - The Perfect Storm

Michael Lundquist

May 19, 2019

1 Introduction

Every programmer is ordained into the church of code with the same simple program. All it does is display the words 'hello world.' In 2014, I took a computer science class, printed 'hello world', and am now forever a member of the church of code. When I wrote my first 'hello world' program in 2014, I was already 21. Twenty One is old for a programmer to write their first 'hello world'. Being nearly a man when I first learned to code has given me a unique perspective on what life is like with and without programming and how fun it truly is. In this essay I will explain: the problems I had before I knew how to program, how much fun I had learning how to solve those problems, and how I currently program with a team of other expert programmers. Throughout this class and in my internship at Leidos this summer, I intend to further research tools that enable team work between developers. I hope to have another 'hello world' moment with some of these tools, and perhaps introduce some of them to my team!

2 before

Before I learned to code I occasionally faced many problems that either required automation or could be performed much faster with code. These problems didn't occur every day, but the problems that I did face heavily influenced and motivated me to keep working hard while learning how to code.

2.1 Joe

The problem that influenced me the most was given to me by my Godfather in an internship I did at his financial advisory firm in the summer of 2013. At the internship, I was hired to copy information from a thousands of websites

and paste the information into an excel spreadsheet. At the time, automating the process didn't occur to me, but the task stuck with me and when I learned how to code, automating repetitive tasks became a very important to me. The task took months of hard work but if I were to do it again today, I would write a web-scraper to automate all the copying and pasting. I would be done with months worth of work in less than 10 hours.

2.2 Business

When I started college in 2012, I was initially a business major. I worked hard as a business major but I ran into certain problems where programming was simply the right solution.

2.2.1 Opportunity cost

In my second semester of college, I took micro-economics. One of the core principles of micro-economics is opportunity cost. Simply put, opportunity cost states that different jobs pay different amounts, so you should choose the job that pays the most. At the same time, I had a close friend, Dimitri, who was a Computer Science major who would talk about how lucrative a career in this field could be. It took me a few years to realize that ultimately my micro-econoimcs class was ironically telling me that Dimitri was right and I shouldn't be majoring in business.

2.2.2 Investing

In high school I worked a few part time jobs and saved up a small amount of money. In the Accounting class I took, in my first year as a business major, I learned the benefits of compound interest and how important it is to invest your money so it can grow. Unfortunately, managing this money became a

bit of a headache because I was always worried if I was investing in the right company. Eventually I came to the conclusion that a computer could help me manage these funds. Years later, I wrote a web scraper to find companies that changed more than 10% in a day that I then used to help me decide which company to invest in.

3 During

In 2014, on a whim, I decided to take my first coding class when I was still a business major. It was completely impractical, as it didn't count toward my degree in any way. Despite it's impracticality, the class could help me in the solve the problems described above. In addition to solving those old problems, new opportunities constantly presented themselves as I was learning to code. It quickly became an obsession that's shaped my life since.

3.1 Online resources

As I was learning to code I found a ton of interesting videos and other resources I could use to apply my new skills. That semester I stayed up long nights drinking beer and watching videos of Defcon (maybe describe what this is) talks alone in my room. Even though I struggled to understand many of the topics in the videos, I dreamed of one day understanding them and advancing in the field. I'm not quite there yet.

3.2 Love interest

One day a friend invited me to a party where I met my girlfriend. She was a political science major and took the same coding class as me, on a whim, like me. She said she was having trouble in the class and I promised I would help her

with it even though I was struggling in the class myself. In a desperate effort to impress her (which I'm still always trying to do), I learned quickly, got caught up in the class, and passed what I learned on to her.

4 After

Learning to code was life changing. After I first learned coding I knew it was something I wanted to do for the rest of my life - the only question was "How?". So after taking that coding class, I switched from being a junior in the College of Business at the University of Mary Washington to being a Freshman Information Technology major at Northern Virginia Community College. Graduating years after my friends is a small price to pay for a rewarding career, financial stability and a girlfriend I love.

4.1 Conclusion - Propose a Topic

These days I'm moving from the academic world of George Mason University to the professional field at Leidos. This summer I'll be returning to Leidos for an internship. In my previous experience at Leidos I learned how to work with developers using certain softwares. When I returned to George Mason, I used this experience to contribute to the Student Run Computing and Technology (SRCT) club at George Mason. Over this summer I look forward to introducing my team at Leidos to some of the technologies that SRCT uses for collaboration.

References

Amdahl, G. M. (1967). Validity of the single processor approach to achieving large scale computing capabilities. In Proceedings of the april 18-20, 1967, spring joint computer conferen (pp. 483–485). AFIPS '67 (Spring). doi:10.1145/1465482.1465560

Dijkstra, E. W. (1965). Cooperating sequential processes, technical report ewd-123.

djna. (2019). Thread isolation in java – stackoverflow. [Online; accessed 16-April-2019]. Retrieved from https://stackoverflow.com/questions/1288154/thread-isolation-in-java

Herlihy, M. (2011). The art of multiprocessor programming. Burlington, MA : Morgan Kaufmann.

Peterson, G. (1981). Myths about the mutual exclusion problem. Information Processing Letters, 12(3), 115–116. doi:https://doi.org/10.1016/0020-0190(81)90106-X

Wikipedia contributors. (2019). Amdahl's law — Wikipedia, the free encyclopedia. [Online; accessed 16-April-2019]. Retrieved from https://en.wikipedia.org/w/index.php?title=Amdahl%27s_law&oldid=890132303