Short and Long Term Goals

In the short term of the next few years, I would like to achieve a position working with a small team to develop my skills as a programmer and a developer. My goal is to be a learner, and absorb as much as possible in the way of learning opportunities. The size of the company does not matter so much as the content of the work, the size of the team, and the environment of the work. I have found that I much prefer an open work environment that allows for easy collaboration. Potentially, I would be interested in pursuing a career in game development. It has always been a dream of mine, but unfortunately the career climate is not optimal. Fortunately, my skill set allows for a range of positions to be entertained throughout the software engineering field.

In the longer term, I would like to find a career position that I could spend a long time at. I am comfortable moving, and would like to be closer to a large city. Again, location is less important that the people. This position I would like to be more comfortable. I could see myself working at a company for several years or decades if the conditions are right. I definitely have the temperament for a leadership position, and I am comfortable with responsibility. Hopefully I don't become complacent, and my plan would be to find a company that has leadership opportunity that I could make into a career.

General Education Electives

SE 409: Software Requirements Engineering

This elective emphasized working with clients to develop a solution to their problems, and was easily one of my most applicable electives. In this course, we studied the Volere Requirements process, and worked directly with a non-engineering client within the university. Our goal was to gather requirements from them, and construct a suitable prototype by the end of the semester. This included a long requirements session, followed up by correspondence to clarify any questions. This class was not necessarily technical, but is easily one of the most important skills an engineer can have. One of the worst sins of software development on behalf of a client or stakeholder is solving the wrong problem with your software. We discussed several considerations, including cultural and social issues that could cause a product to be scrapped, and how to avoid them.

Com S 437: Multimedia and Game Programming

This course was a fun one, but also one that gave me a huge appreciation for game development. It had been on my radar since I was a freshman, and taking the class was a lot of work. The class emphasized being clever in our code; the limitations of hardware often mean shortcuts and "hacks" are used to achieve efficiency. Because of this, our class also discussed

content, particularly in fairness and ethics. It is important to consider the player and any frustrations or biases they may have when playing your game.

SE 329: Software Project Management

In this course, our team of four took turns being project manager on four different courses. One of the most important skills an engineer of any trade can have is collaboration. This was met with variable success, and I learned a lot about working with others in a challenging way. Our group did not always cooperate, and two of our members in particular seemed to want to work independently. What I learned was that work environment is important, and not all colleagues will work well with each other. Sometimes this can be overcome, and sometimes personalities are too strong to cooperate. Even though the class and my team members were difficult at times, the experience was valuable in learning the type of person I am comfortable working with.

We did however in this course explore a number of socioeconomic issues. From a technical standpoint, we had an application that could analyze a user's last 200 tweets and use IBM's Watson to provide textual analysis. From a social standpoint, it definitely showed how easy it was to take a person's online ramblings and provide data about it. Equally though, it's almost a cautionary tale about Big Data; is this information relevant? Should this information be available? How accurate is Watson, and *should* we draw conclusions from its data? For the purpose of the class, we did not attempt to answer these questions, but it is certainly something we as engineers should think about when we are making software that will be available to the public.