Memorandum

Date: October 9, 2016

To: Michael Knievel, Professor for the Department of English

From: Caleb Carlson, Computer Science Student

Subject: Workplace Writing Memo

Each workplace has their own need and use for technical documentation. This is a memo containing the basic information gathered in an interview for technical writing in a computer science environment for a university IT department. The department of the person who I got my information from is in charge of putting together programs for university employees to gather information about students. They work with University of Wyoming databases to store, query, and gather data.

One of the first and foremost forms of communication in a software company and the place I did my interview is via email. Information about new requests, testing programs, and general workplace communication are all done via Outlook, the University’s email client. Being able to craft professional emails as well as quick on-the-fly-emails is important to success in the department.

In the computer science field, a large part of the communication done within the work place is done through clients. When creating or writing programs, a client usually knows what they want done, but not necessarily how a particular solution to the problem might work. As a result, it requires the programmer to put together a piece of documentation on how to run the software. The challenge with this is that often the programmer understands the software and its inner workings far better than that of the client. While, like any workplace, there are managers who can step in when need be, it is each programmer’s job to keep in contact with their client and provide them with the necessary information they need. It takes a thorough explanation and sometimes careful wording to make sure that concepts are properly explained. If a programmer is not careful this can often lead to a confused and frustrated client. This also draws out the testing process, which wastes time that can be better spent on improving the program or working on another piece of software.

Client documentation is not the only form of communication for the company I looked into. Within the University’s database software department, there is a need to document specifics about programs and what data is being used so this is done through “in-house” documentation. This is, as the name suggests, a documentation that is strictly meant for people inside the University’s software department. This will usually be used to help another programmer, who is new to a project, catch up on how a specific piece of software works before they begin or speed up the process of locating what data might be modified by a specific program. Any in-house documentation is created purely for programmer’s eyes as the document may contain information that could be considered sensitive. Besides that, the client documentation is created with the idea that it is a “cleaner” version of the in-house version. That way a client does not have to go digging through a huge document to find a small section of information they need. A client’s primary job is not to test the programs that are created so it is important that any documentation created for them is easy to understand and use. This way they can test whatever they need, approve of the programmer’s work, and get back to their regular tasks quickly.

Presentations are also common among the workplace. “Demos” of client software or modifications are often set up as a part of the testing process. This offers a more personal experience as opposed to written documentation and allows for questions to be answered faster. Usually, when a meeting is set up, certain test cases are chosen to be shown for a specific program. This eliminates any extra time in testing and the programmer can show a client that the software works for any specific cases they wanted to look out for. Once the demo is over clients are allowed to ask questions and provide any further cases where the program needs to be tested. These meetings can be set up between just the programmer and the client or a few programmers depending on how complex the request from the client is and how many programs are ready to test. It is not uncommon to see two hour meetings where several programmers meet up to present their changes or new programs to shared clients.

One of the best and most common used techniques when preparing to create documents and presentations is testing. Testing frequently and often allows the software creator to work out all of the “bugs” in a program. It is highly frowned upon for a programmer to give clients a presentation on a software that they are unsure works. That is not to say it is bad for a client to come back and say the program is not behaving as intended because that happens on a daily basis. What programmers do is try to prevent the client from testing the same program an excessive amount of times. In the case of clients, the less testing the better. It is the programmers job to make sure the testing phase goes quickly and efficiently for the client. It is not professional to have the client test it and come back with errors multiple times.

Another technique that is less common to use is flow charts. These are diagrams that help guide a use through how the program works and what decisions the program makes. For visual people, like myself, it helps break down the complexity of the program into a semi-linear format to see. To assist with the creation of documents and presentations, the material is sometimes shared with colleagues to help edit and polish the program further. If a colleague who has no knowledge of the subject can get through the documentation without issue, then it is more than likely that anyone else viewing the presentation of documentation can understand the program as well.

A lot of the documentation put together is done towards the end of a project so there is not a ton of research put into the documentation. If any additional information is required to write the documentation, besides information needed to complete a program, it is usually found via a search engine, like Google, on the web. One of the key skills to becoming a better programmer is to know how to use a search engine quickly and efficiently. This is key not only because it saves time in getting to information that you do not know, but it also helps keep the programmer on top of new information in a field that changes quickly with the times.

As suggested to me in my interview, the best way to get better at documentation and presentation in my field is to practice. There is a learning curve with any new area and it is important to keep practicing and be open to correction to improve upon work. My interviewee suggested that applicants for a position in the field aspire to have excellent communication and written skills.