Garrett Capaccioli

**EX02 Assignment**

**EX01**

1. Public. Software engineers shall act consistently with the public interest.

Public interest and corporate interest are constantly at odds with each other. Consider this situation, you work for a company that develops a social networking application. Your boss wants you to incorporate a feature that will grab and analyze users personal data in order to deliver them targeted advertising. This change would allow the company to become more profitable and increase your pay. However, it would also compromise users data for the sole benefit of the company and their potential advertising partners.

Morally speaking I would not have a problem incorporating the feature into the app as long as the users were well informed about how their data is being used. Users are using the app free of charge and it is unrealistic to expect a company to operate without any sort of compensation. So while it may not be directly in the publics best interest, overall it is reasonable and fair to incorporate such a feature.

2. Client and employer. Software engineers shall act in a manner that is in

the best interests of their client and employer, consistent with the public

interest.

A client wants an additional feature added to the application that you are currently getting ready to ship. However, adding said feature would end up costing you and your employer significant amounts of time and resources that you cannot afford. Adding the feature would be in the clients best interests but not adding the feature would be in the best interests of both you and your employer.

In this situation, I would most likely decline the clients request for a new feature. Since we are close to shipping the cost of adding a feature would be very high and most likely require (depending on the feature) additional refactoring and work. Unless the client was prepared to front the additional cost of the new feature, adding the feature would be too costly to incorporate. I personally don't think it is entirely possible to serve both your clients and your employers interests at the same time, but rather compromise on what can and cannot be done.

3. Product. Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

Your company is designing a photo editing application and there are several bugs in the software. Your boss tells you to ship the software with the bugs, since your competitor is releasing a similar application as well. If you ship the software you are releasing a less-than-perfect piece of software, but if you don't ship you could lose a significant market share of potential customers.

The decision to ship the software or not would depend heavily on the severity of the bugs. If they were bugs that did not occur often and were not very costly to the user, than I would ship the software. However, if the bugs were severe and did prevent effective use for the user, I would decide to wait. While the company may lose out on some market share, it would be worse to ruin the companies reputation with a buggy product.

4. Judgment. Software engineers shall maintain integrity and independence

in their professional judgment.

Your boss wants you to give your stamp of approval for a program your company has developed. However, the code is known to have a vulnerability that could compromise the users whole computer if exploited. If you give your approval the product will ship and be distributed to thousands of users. However, if you decline you could be fired or at least displease your boss.

I would not give my approval for the software to be shipped because of the potential cost to the users. Having a loyal user base is an extremely important asset for a company and knowingly compromising the users computers would be very morally unacceptable. Knowingly compromising users is not what a programmer should do if they are able to do something about it.

**EX02**

While I have not worked on many large group projects, the few that I have worked on have used a Democratic organization scheme. Since most of the projects that I have participated in have been school oriented, working with peers instead of a boss/employee hierarchy was fairly standard. The teams that I have worked with were typically no more than 3 people, so we often used a modified version of the Agile model. We had no team leader, and no product owner (save for our professor) and relied heavily on small group communication and splitting up of tasks. We were following an XP practice, so within our group we had all of the necessary skills required to complete the project. I would have classified each of us as a generalizing specialists, since we each required a broad range of knowledge in order to complete our project. Overall,i would say our organization served us well for our intended purposes. Since we were not in a company and did not have any client the democratic delegation of tasks seemed to be the most effective. However, a significant downside to this form of organization was the fact that it relied heavily on individual responsibility within the group. This means that if a particular member of the group failed to achieve their promised goals, the entire group would suffer in turn. Despite this, I still affirm that the small Agile grouping was the most effective organization we could have mustered.

**EX03**

There are many different groups of people that have the potential to work for Microsoft however, there are a few that probably should not. One of the main criteria for working with Microsoft is the ability to function as part of a team. Those that have the inability to work with other's code, or cannot work on a group project will fail to meet the Microsoft standards for being an employee. In addition, those that cannot describe their work will not succeed at Microsoft. Since Microsoft's development model relies heavily on documentation and a clear understanding of the code being written, someone who fails to clearly explain their code would have a difficult time working for Microsoft. Thirdly, Microsoft requires that all employee be able to work within very specific time constrains with carefully designed schedules and time frames for each feature. Those who are disorganized and not timely with their code would have a hard time succeeding at Microsoft. The kinds of people that would struggle at Microsoft would be those who are independent, bad at communicating their code in simple English and who like to work in a very flexible time frame. While they may be decent coders they would not fit well with the mentality and the overall structure of working for Microsoft.