Donovan Moini
Dr. B.J. Johnson
CMSI 401
10/2/2019

Assignment 1: ValuJet Article Summary

On May of 1996, ValuJet 592 crashed, resulting in over 100 deaths. There are three types of airplane accidents this could fall into. The first is procedural, meaning single, obvious mistakes. The second is engineered, which includes material failures which should have been easily predicted or discovered. The third, which the ValuJet 592 crash falls under, is a system accident, meaning a combination of complexities occurring simultaneously. During the flight, ValuJet 592 caught fire, which engulfed the plane in flames and caused the crash. Upon crashing, a recovery team discovered that the flight had held over 100 oxygen generators, which caused the plane to catch fire. Additionally, the cargo hold that contained these was unequipped with fire detection and extinguishing systems and contained many other flammable objects.

Before the crash, workers had stored the oxygen containers into unmarked cardboard boxes and did not place the required safety caps over the firing pins, despite instructions from ValuJet. Additionally, ValuJet 592 took the boxes despite federal regulations prohibiting them from doing so. Even more so, an internal report was written recommending that ValuJet be grounded and restarted. But that report was not found until after the crash, which led to ValuJet being grounded indefinitely.

Charles Perrow theorizes that “the control and operation of the riskiest technologies require organizations so complex that serious failures are virtually guaranteed to occur. Those failures will occasionally combine in unforeseeable ways … and spin out of control” (Langewiesche 13). Perrow also notes “interactive complexity,” meaning elements are linked in

multiple and unpredictable ways. If a system is large, this results in an infinite combination of links. Ultimately, from the combination of unprovided safety caps, unclear instructions due to "engineerspeak," unlabeled boxes of oxygen containers, and more, this combination led to the crash of ValuJet 592.

Even though this situation primarily resulted in hardware malfunctions rather than software issues, many lessons and aspects from this situation apply to software development. The two I would like to discuss in this assignment include Perrow's idea of "interactive complexity" and the importance of communication within organizations. The first one that I would like to touch on relates to Perrow's idea of "interactive complexity." According to "interactive complexity," the involved elements are linked in multiple and unpredictable ways, and the failure of one part may lead to the failure of another part. With large systems especially, this will lead to an infinite combination of possibilities. This can be seen in projects with large tech stacks. With so many services working with one another, it is impossible to know all the combinations of how they will work with each other, either correctly or incorrectly. While this may be a small issue such as a calculator application crashing when inputting a specific combination of buttons in a specific order, this can be a larger issue such as incorrect data being sent back and forth with a banking database due to too many people simultaneously using the system. Of course, we have people such as those in quality assurance who test for such issues to avoid them from happening in the final builds, but they can only test for what is limited by what they think they need or know to test for. As Perrow implies, there is an infinite number of combinations of links for large systems, so no person or group could ever test all possible scenarios within the system. This is, unfortunately, the reality and there is no explicit solution to

finding infinite issues in a system, but when issues do arise, updates and patches should be released frequently and consistently to address the issues that were not found during development.

The second aspect I would like to touch on is the importance of communication between levels of work within a project or organization. In the ValuJet example, a large factor to the crash came from a lack of communication between the levels of operation. This included the workers not alerting that they did not receive safety caps for the firing pins, unclear language distinguishing expended from expired, labelling the boxes containing the oxygen containers as "aircraft parts" incorrectly, forgetting about the internal report recommending that ValuJet should be grounded and started all over again, and much more, leading to the death of over one hundred individuals. While maybe not as fatal as in the airplane industry, this is just as important within software development when you have people working both at different levels and people working across the world remotely at the same level. Consistent and clear communication is crucial when working with so many different people on the same project to ensure its success. An example could be a manager vaguely presenting a problem to the software developers, all of whom see the problem from different perspectives. Software development is incredibly collaborative, so the software developers will need to be on the same page with each other about the problem. If the software developers do not communicate with each other to get on the same page though, this can lead to issues in how they solve the problem as a team. Once they communicate with each other and get on the same page, the problem can be correctly worked on by the software developers.

Unclear communication issues can also be seen with different departments working with each other due to a lack of understanding technical terms, just like the workers not understanding expended from expired. Within every field, you learn technical terms within that field, but those terms are not naturally known by people outside of this field. For example, a software developer will probably not know the specialized terms used within the marketing department and vice versa. But bridging that gap between the two fields to have a common understanding is crucial when working in groups with multiple different departments. If a manager is presenting to a group of possible funders who are not software engineers, and possibly not fully versed in software engineering terminology, the manager would need to find a way to describe the software in terms that they would understand. If the manager is unable to do so, the loss of funding the current project is a possibility since people will usually not fund something they do not understand. Communication is typically an important skill in any workplace and industry. Poor or a lack of communication rarely results in death, but unfortunately in places such as the airplane industry there is a possibility for such, and the ValuJet 592 flight is an example of poor communication having a fatal result.