**ValuJet – Flight 592**

**Synopsis** The introduction of the article gives multiple accounts of the ValuJet 592 accident. A fisherman’s first-person report and dispatcher transcripts of the accident lay the ground work for what happened that afternoon in 1996. The author defines the 3 kinds of plausible accidents—procedural, engineered, system. The ValuJet accident was a system accident, which is the result of confusion within the management of our technologies. The account of the traffic controller shows the final moments of communication before the accident. The article continues with the controller’s efforts to guide the plane to safety, the immediate recovery operation, and the scramble to find the cause of the accident. Eventually it was discovered the smoke in the cabin originated from expired oxygen tanks that had combusted in the cargo of the plane. Many different parties are at fault for this negligence—SabreTech, the FAA, and ValuJet executives. SabreTech was contracted to remove the old, dangerous oxygen containers from the plane, ensuring all regulations were met and procedures followed. However, mechanics signed off on the job without placing the unique safety caps on the canisters’ firing pins. SabreTech kept the empty, combustible canister in their warehouse for some time. SabreTech had a potential client visit scheduled at their warehouse and wanted to remove any waste or clutter. SabreTech decided to sign off on shipping the empty canisters back to ValuJet because they didn’t want them in their space and they were technically owed by ValuJet. The canisters were loaded on to ValuJet Flight 592, a gross violation of the regulations. As the plane gained in altitude, higher pressures caused the expired oxygen tanks to combust. The fire spread throughout wiring, computers, and controls, eventually causing loss of control and the accident. This accident “fits the most basic definitions of an accident caused by the very functioning of the system or industry within which it occurred. Flight 592 burned because of its cargo of oxygen generators, yes, but more fundamentally because of a tangle of confusions that will take some entirely different form next time” (W. Langewiesche). Although the action of signing off is functional and necessary it becomes dysfunctional when people don’t do their jobs to the fullest. One fault in the system, one slightly flawed employee can result in disaster.

**Thoughts/Reactions** My first reactions to this article were disbelief of the detail of the first-person account and unsurprised that this crash was such a mystery at the beginning. The article pulls you into the story immediately with the details of the crash itself followed by the ensuing mayhem to solve this mystery.

My following reaction was the number of time this accident could have been prevented without any more action required than what is detailed in each person’s job description. If the employees were to pay closer attention and perform their jobs at their highest ability then ValuJet crash would not have happened.

One thing to point out in the dialogue between the pilots and air traffic control is the null read out, “XXX”, for the plane’s altitude. Certainly, in 1996 the technology in airplanes had to be some of the most sophisticated but in comparison to now it is archaic. This null read out is a result of the latency of the wireless systems and computers of the past. Nowadays, planes are extremely accurately tracked—even from a smartphone.

The central cause of this accident is not the canisters themselves combusting, but actually the numerous employees’ lack of attention to detail. Company executives and presidents expect the absolute best out of their employees and in return the employees should want to work their hardest. Not only is the idea of ‘attention to detail’ extremely applicable in the field of computer science, but even more so in the greater scheme of software development. Beginning with the conception of software, product managers must provide accurate and complete details of the acceptance criteria. From there, the developer must closely choose which area of code to work on or where to create a new directory or file. Next, the developer must craft code in a way that it doesn’t interfere with anything it shouldn’t but still meets the detailed acceptance criteria. Soon after, the code must be closely reviewed by a peer. Finally, the quality assurance team must test every detail of the software to ensure proper functionality and that no bugs exist.

A recent scenario at work elaborates on the theme of ‘attention to detail’. I was assigned a task to implement a hide or show password button on all the login pages across the site, given the acceptance criteria, and got to work. After finishing up, I checked that my code met the acceptance criteria and sent it in for code review. Soon after, I got a Slack message saying I needed to make some changes and that the acceptance criteria was updated. I learned that the acceptance criteria was incomplete, which resulted in a minor accident—wasted time. Although much smaller scale than an airliner crash, this is an example of how important attention to detail is.

Nowadays, it is vital to consider security one of the top concerns when building software because certain vulnerabilities can cost a developer or company everything they have. Developers ought to use ‘best practices’ when writing code and making design decisions such as host servers and account authentication. Security is not a factor in our senior project because it is a closed system; however, attention to detail and testing will be very important.

We will need to carefully test our software for multiple things. First, realistic simulation must be, well, real! This testing will include trial and error in the beginning until the mechanisms are fine tuned and under our complete control. Next, we will have to test that our software accurately collects data. Data acquisition is central to this research, so we must ensure our software provides the researchers with exactly the data they want. Finally, as our software is being create we will document each piece: how it works, how to use it, how it was built, and any upkeep.