Amelia Jay

10/2/19

CMSI 401

ValuJet

In 1996 ValuJet Airlines Flight 592 crashed into the swamps of Everglades Holiday Park resulting in the deaths of all who were on board. This included two pilots, three flight attendants, and 105 passengers. ValuJet, a discount airline based out of Atlanta, had a history of a few machinery malfunctions, but no one was prepared for a disaster quite like this.

The article in the *Atlantic* went over three different types of airplane accidents. There are procedural, engineered, and system accidents. Procedural accidents result from single and obvious mistakes; the reason for the accident can be immediately understood. These accidents are avoided when pilots, mechanics, ramp agents, and air-traffic controllers observe simple rules (like don’t fly into violent thunderstorms, for example). The second type of accidents, engineered accidents, consist of materials failures that should have been predicted by the designers and test pilots. These accidents, although hard to understand at first, result in examination and tangible solutions. The last type of accident, which was the type that ValuJet experience in the 1996 fatal crash, is called a system accident. It is hard to truly understand these accidents, but ultimately, the fault lies on the entire airline system. Things went wrong in multiple places as the article says: “In this case the organization includes not only ValueJet, the archetype of new-style airlines, but also the contractors that serve it and the government entities that, despite economic deregulation, are expected to oversee it. Taken as a whole, the airline system is complex indeed.”

The system accident that ValuJet experienced occurred in the following way. Flight 592 was carrying oxygen generators that came from three MD-80s (a modern kind of twin jet that ValuJet had recently acquired and was refurbishing in Miami) in its forward cargo hold. Instead of being properly secured, the firing pin on the generators was just taped shut. These generators hold a complex chemical reaction and when they’re generating they heat up (upwards of 500 degrees fahrenheit). The plane’s cargo hold is build airtight, so if there is a fire, it would suffocate it. For that reason, there were no smoke alarms in the cargo hold. Unfortunately, the oxygen generators were in the cargo hold, and during take off at least one of the generators started filling the whole compartment with oxygen which fuels fires. The compartment got super hot because the chemical process is exothermic and everything down there caught fire. The fire burnt through important cables that allowed the pilots’ controls to actually control the plane. The plane went down and shattered on impact shortly after the pilots lost control.

There were clearly vulnerabilities in the system that got ValuJet into this position. Vulnerabilities can occur in software engineering as well. Flight 592 could have gone down in other ways. For example, if the plane’s software had been hacked from vulnerabilities in the code, the pilots could have lost control that way. But that didn’t happen. What happened was a series of design and communication failures that combined and resulted in this tragic crash. First of all, in designing the cargo hold, there was no smoke alarm. Of course, the designers must have believed that there wouldn’t be a case where oxygen would be generated in the compartment. However, designers must keep edge cases in mind: what *if* oxygen were to be present in the compartment? An alarm should have been put there to alert people that a fire has started in the cargo hold. As software engineers, we need to keep in mind the design of our programs as well. What *if* a certain edge case were to occur? We cannot rule out possibilities that seem unreasonable, especially if the result of those possibilities occurring is detrimental to the users or the product or code.

Another issue with this system accident was the communication across all parties involved. The oxygen generators should have been properly sealed, but the consequences of not doing so were not known amongst the people who just taped them. Similarly to ValuJet, lack of communication between engineers and other team members of a software team can happen and can result in negative consequences for the product being developed. For example, if one engineer sees something they don’t recognize in the code and think it isn’t important, they might move it or just get rid of it. This is not good practice in software development, but it definitely can happen.

A good way to encourage good communication is for teams to have daily stand up meetings where everyone checks in and explains what they’ve been working on. The Agile development with sprints and sprint planning is really great too because everyone knows what their job and their team members’ jobs are. If working with other teams on a project it is essential to have frequent meetings to make sure you are on the same page and that each person understands what the goal is and what the key things to keep in mind are.

The “lessons of ValuJet” absolutely apply to the software discipline. In some cases, software mistakes can result in fatalities, but there can also be other devastating consequences. Information that belongs to individuals, like their social security number, their financial information, and their passwords, can be accessed if engineers haven’t done their job correctly. People’s information can also be lost or deleted if companies accidentally delete a database, for example. Ethically, if a software development or engineering company makes a promise of safety to its customers, they must fulfill that promise. ValuJet made the promise to its passengers that it could get them from point A to point B safely, and they failed. It is the duty of software engineers to do their best to prevent failures of promises about individuals’ security and safety.

Specifically applying this to my life, I am planning to take the lessons of ValuJet very seriously while working on my senior project. The scheduler that my team is creating is going to be scheduling different individuals. These individuals will have their information stored in the database so that they can be identified. We will need to be sure that any personal information (like phone number or email) that the admin may want stored in the database is kept secure.

Source:

<https://www.theatlantic.com/magazine/archive/1998/03/the-lessons-of-valujet-592/306534/>