ValuJet flight 592 was a tragic disaster. However, one way we can honor the people who died, is to learn from this incident and try to do better. In preparation for writing this, I read *The Lessons of ValuJet 592* by William Langewiesche in the Atlantic. In this paper I will give a brief overview of the events that led up to the crash of ValuJet 592, an explanation of what I believe the underlying problems of this situation to be, and how they can relate to the world of software development.

On May 11th 1996 ValuJet flight 592 crashed into a Floridian swamp. After the initial cleanup crew found the black boxes that record the flight data, investigators began to piece together the story for why the plan had crashed. Eventually they found their answer. ValuJet was an extremely affordable airline. They thrived off cutting corners wherever they could. This caused a workplace that focused only on pure profit and not safety to be born.

ValuJet asked SabreTech to do some maintenance on their planes. Specifically, they wanted to replace the oxygen canisters that provide oxygen to the passengers in emergency situations. First, some mehanics pulled these canisters out of the plane, and then tagged them. Unfortunately, they tagged them incorrectly as “Repairable” which becomes important later. Also, these mechanics did not put a restraining cap on like they were supposed to. Then these canisters sat for a few days before getting moved to the shipping area.

Once in the shipping area, the clerk decided to send these canisters back to ValuJet headquarters since they were marked repairable. He packaged them up in a cardboard box and asked his supervisor to sign off on the shipping receipt. At this point, the supervisor signed off saying the canisters had been packaged correctly. However, they did not have the needed caps and were stored in the wrong positions. Also, after the box was wrapped up, both the clerk and the supervisor neglected to attach the required “Toxic Materials” tag onto the box. After making it past this packaging process, they were sent to the airport for transport.

Again, we see similar breaks of protocol at the airport during shipment. These oxygen canisters could contain trace elements of toxic residue according to federal regulations. They should have not been able to get on a flight without special exception. However, both the ramp agent and the pilot failed to recall this fact and accepted the canisters onto the flight. They were put into the caro hold, next to multiple rubber tires.

This series of unfortunate events led to the flight taking off with improperly stored and labeled oxygen canisters. The most likely result based off the flight recorder and communications with the flight tower was one of the canisters accidentally ignited. This lead to the other canisters heating up and igniting stating a very hot fire. The cargo hold did not have an anti-fire safety precaution. The tires in the hold also added to the fire which eventually burned through critical electronics. The pilots lost control of the airplane and crashed.

In the article mentioned at the beginning of this essay, the author exemplifies multiple points in this process where events could have changed minorly and the entire situation could have been avoided. Since the writer himself was a pilot, he was able to analyze the process in terms of flight management. However, there are a few overarching problems that I would like to point out.

First, the requirements for safety are far too complicated and confusing. This led the workers to take shortcuts. For example, the maintenance workers that took the canisters out of the plane did not put the caps on the canisters. This is likely because they did not know they were even supposed to. As demonstrated in the article, the manual that explained if these caps should be used is extremely long and even contradicts itself. If the safety process was more streamlined, they may have been more inclined to put the caps on and avoided the disaster entirely.

Second, the oversight system was very flawed. When the workers packed up the canisters, they had to get multiple supervisors to check their work. Multiple people signed off saying the package was

okay, even when it clearly was not according to the very checklist they initialed. At first glance, it seems the companies acted accordingly here by having multiple “checking” systems, but the error was none of the systems were effective. Part of this is due to how the current flight system encourages cheapness and therefore corners will be cut. If the focus was instead on safety not simply money, the supervisors might have felt more empowered to do their job correctly.

Finally, the final actors were not experienced enough to handle the situation laid before them. The author of the article claims that the ramp agent and the pilot had been formally trained in how to recognize potentially toxic materials. In this case, they failed to do so. I believe this is due to lack of knowledge. They may have been formally trained on WHAT to do but not WHY they do it. Most likely, they had not needed to check for hazardous materials before so simply were not checking for it this time.

These three problems can be examined in software development. For clarity, the problems are: confusing and complicated instructions, focus on a dangerous goal, and lack of true knowledge by the people closest to the product. I will explain exactly how each of these pertain to software development.

First, tutorials and explanations on how to use a product have to be simple and effective. IF they tutorials are too complicated, the users will simply not listen to them, which can lead to user error down the line. If a user is not using the product correctly, the application may seem broken or ineffective. Software developers should ensure the product they are building will be easy to use for their target audience.

Secondly, software developers need to focus on the goal of the project. This goal has to be clearly defined to all developers. If the focus is simply a quick cash grab, the application wll most likely come out “half baked” and inefficient. The best focus in my opinion is to make your product be as helpful as possible. Note that sometimes, in order to be more helpful, more time may be required in

development. A perfect example of this is the video game industry. Recently many games have been released because the management at the company said it was time, not because the game was ready for production. The consumers can see this and the application will ultimately fail and therefore be unhelpful, as opposed to spending time to polish the game which may last for decades.

Finally, the people who interact with the product right before it gets released to customers have to have true knowledge of the application. Software development has many stages and possibly many substages. If every system works great but does not work together, the application will ultimately fail. There needs to be inspection of the application as a whole, but this inspection also needs to understand each system. They do not necessarily need to know the subsystems but they need to understand why each system works the way it does. This will allow them to more effectively judge if the application is fully coherent.

ValuJet flight 592 was a tragic disaster. If we can study the processes behind the disaster and use them to help better understand our own processes, perhaps we can make at least some light of the situation. I have drawn the connection from the airline industry to the software development industry, but these concepts can be molded to fit many more industries.