Alvin Lai

10/2/2020

Summary: ValuJet

ValuJet flight 592 crashed into the Everglades park in 1996 and was shrouded in mystery. While one expects a plane crash to leave behind a large, burning pile of rubble as well as a tall plume of thick black smoke, this crash only sent dirt into the air and scattered the belongings of the passengers around the site. The circumstances surrounding the crash begs the question of who is to blame for such a tragedy. Was it the result of a careless technician, engineer, flight attendant, pilot, or air traffic controller? Or could the situation simply be a result of uncontrollable, unforeseeable circumstances? The article suggests that the crash resulted from carelessness and negligence that trickled down the development chain all the way from the FAA itself.

The FAA at the time was preoccupied with other commitments such as counterterrorism, leading them to only assign three inspectors to ValuJet. In the meantime, ValuJet cut as many corners as they could to maintain an affordable and appealing cost for its customers. ValuJet paid their flight attendants and other employees poorly, hired cheap contractors which neglected rules and regulations, and failed to properly train their employees. Such a slew of carelessness and negligence allowed for many unacceptable choices to be made which unfortunately resulted in the deaths of 110 individuals.

While it may not be immediately apparent, the ValuJet failure can teach us a lot about software engineering projects and how critical it is to follow best practices, principles, and standardized approaches. This disaster reminds us of the importance of documentation and communication just to name a few key components of software development, and how they are all interconnected to produce a strong product.

Software engineering projects begin on the drawing board to figure out the needs and requirements of a product. For projects of any scale, it is important to have a mutual understanding of what the product is, what it needs to do, how it will accomplish those tasks, and so on. Creating and constantly maintaining documentation is a simple yet effective way of communicating the idea of a product, as it can help reinforce people's understanding of the product or create mental models of it.

In ValuJet's situation, inconsistent documentation played a major role in the project's cascading failures. From the article titled "The Lessons of ValuJet 592" by William Langewiesche, it was stated that, "two unfortunate mechanics… signed off on the nonexistent safety caps…," which indicates that there was a lack of care for documentation on the project. Project managers may have deliberately chosen to disregard documentation due to time constraints, a lack of resources, or simply a negative attitude towards the mundane and meticulous process. Regardless, normalizing the dismissal of proper documentation allowed project participants to develop a false sense of commitment. As Langewiesche put it, improper paperwork contributed to "the creation of an entire pretend reality that includes unworkable chains of command, unlearnable training programs, unreadable manuals, and the fiction of regulations, checks, and controls."

Having done some part of Synth Trainer's documentation, I can understand why the individuals who worked on ValuJet failed to generate sufficient documentation. For a relatively small project, the process has already been quite long and tedious. Coupled with assignments for other classes, job searching and other aspects of life, sitting down to meticulously document all components of Synth Trainer may not seem like a high priority. I have questioned the documentation step many times wondering why it’s necessary to do it in such detail when I could allocate my time and energy into programming the actual product itself. However, it was only after finishing a draft of the documentation that I realized its significance in the process. My team and I gained the ability to develop a concrete mental model to build off of and we all gained an understanding of what we are trying to accomplish. This will allow us to have a plan going into the project and have a better idea of our progress along the way. The documentation is also valuable in that it will allow new potential customers, users, and developers to get a deeper understanding of the product in case they want to make modifications of their own or build upon the existing product. It is important for individuals to remember that documentation is not final and that it should be constantly updated to account for new additions, weak points, and so on.

Langewiesche also touched upon miscommunication among engineers who worked for SabreTech and ValuJet. With teams comprised of both Spanish and English-speaking engineers, another layer of complexity was laid on top of an already complex set of circumstances. The language barrier between Spanish and English-speaking engineers ultimately led to an amplification of problems that were already large. Training on these types of issues could have resolved and prevented many of the problems that ValuJet and SabreTech engineers faced, but the prioritization of cost over quality on ValuJet’s end only worsened the issues.

Miscommunication can occur in many ways and I admit that I still have a lot to learn when it comes to expressing ideas and trying to help people understand a vision. When considering what I have to improve on in my communication skills, these are some things I often take into account: people’s experiences, why someone might feel a certain way (afraid to ask questions, afraid to be assertive, etc.), and the clarity of my communication. I think tackling communication issues goes far beyond simply breaking down barriers between languages and I think it is something beneficial to keep in mind for the Synth Trainer project and any projects in the future.

I believe these two components of software design (documentation and communication) already encompass a wide range of issues faced by software developers. A strong mutual understanding in a product alongside thoughtful and open communication creates a stable foundation for teams to build upon. It's unfortunate that ValuJet's lack of consideration for these design components costed the lives of 110 people, but it would be an even bigger tragedy if we did not take this opportunity to learn from the lessons of this event.