CMSI 402 Semester Assignment - ValuJet, System Failures, and Computer Software Engineering

Andrew Won

March 14, 2013

1 "The Lessons of ValuJet 592" by William Langewiesche

There are always mistakes and problems that may arise in the course of engineering, but few things are as tragic as when lives are lost as a result of avoidable mistakes. "The Lessons of ValuJet 592" by William Langewiesche follows the crash, the investigation, and provides insight that might be gained from ValuJet 592, a twin-engine McDonnell Douglas DC-9 that crashed on May 11, 1996, killing all 110 people on board [Langewiesche, 1998]. By reviewing the nature and origin of past problems which led to this accident we can learn lessons to apply to our future software development.

The article begins with a description of the crash by Walton Little, an observer who happened to be fishing near the site of the crash. Mr. Little reported that, "There was no smoke, no strange engine noise, no debris in the air, no dangling materials or control surfaces, no apparent deformation of the airframe, and no areas that appeared to have missing panels or surfaces," before the plane flew straight into the ground. Little's account of what was not present is significant because it rules out many of the typical explanations involved in plane crashes. Without any evidence of an engineered problem it was at first difficult to comprehend why ValuJet 592 had flown straight into the ground.

Langewiesche remarks that even with a reason for the crash having been found and actions taken by ValuJet and the FAA, it does not mean that the problem is truly resolved. The reason why a true resolution is difficult to find in the ValuJet 592 crash is because of the nature of the accident. Langewiesche breaks airplane accidents into three types, as outlined in the table below. The first two of which are natural to think of, but the third of which is the elusive trouble-maker that is difficult to address and that some argue we cannot avoid in the future.

Types of airplane accidents	
"Procedural"	Result from single obvious mistakes
"Engineered"	Materials failures that should have been predicted by designers or discovered by test pilots. Usually defy understanding at first but yield to examination and result in tangible solutions.
"System"	Most elusive. Charles Perrow, a Yale sociologist, calls these "normal" accidents because they are the "bastards born of the confusion that lies within the complex organizations with which we manage our dangerous technologies."

The National Transportation Safety Board (NTSB) quickly arrived on the site after the crash and began their investigation. Shortly after beginning the investigation they realized that there may have been an explosion resulting in hazardous materials, and they may have even realized that it may have originated in the forward cargo hold, where there was no fire detection or extinguishing systems. Paperwork indicated that the forward cargo hold had been loaded with ValuJet "company material," which on this incident happened to be a combination of three tires and five cardboard boxes of old oxygen generators. This may have raised some alarms because oxygen generators should not have been on this flight.

The oxygen generators found in the forward cargo hold of ValuJet 592 were chemical oxygen generators removed from planes that ValuJet had been renovating. These chemical oxygen generators were commonly used in aircraft to provide oxygen to passengers in the event of sudden loss of pressure in the cabin of an aircraft. In the event of a sudden pressure loss, face masks would fall from the ceiling of the plane and passengers could begin a flow of oxygen to the masks by pulling on the tube, which would trigger the chemical process by which oxygen could be generated and delivered to the passengers. The danger with these particular oxygen generators were that they were exothermic- they generated heat outside of the containers, and were a fire hazard. This fire hazard was a known danger that was not treated properly.

The danger posed by the oxygen generators started when mechanics working for SabreTech, a company routinely hired by ValuJet and many other airlines to do maintenance and work on airplanes, were renovating three MD-80 airplanes recently purchased by ValuJet. Mechanics working for SabreTech removed the oxygen generators from the airplanes and placed them into cardboard boxes, but failed to place the required plastic safety caps over the firing pins of the oxygen generators. The lack of safety caps meant that these exothermic oxygen generators could inadvertently fire and pose a threat of fire. According to the article, the reason why the safety caps were not used was because "no one had such caps, or cared much about finding them." SabreTech had failed to provide the necessary materials to do the job properly, but the mechanics signed off certifying that safety caps had been placed properly anyways.

The oxygen generators that had been placed into cardboard boxes sat around at SabreTech for awhile without really being dealt with. When they finally made their way to the shipping department at SabreTech, a manager noticed them and told a shipping clerk to get rid of them to make sure the site was clean for an upcoming inspection from a potential client. The shipping clerk secured the boxes and then asked the receiving clerk to make out a shipping ticket with "oxygen canisters –empty" written on them. They then sat around for another couple days. Finally, on May 11, 1996, the boxes containing the oxygen generators, now incorrectly marked as being empty, were driven to the airport for ValuJet Flight 592.

When the oxygen generators that SabreTech had removed were driven to the airport, the ValuJet ramp agent defied federal regulation and accepted the shipment. ValuJet was not allowed to carry any item that contained hazardous materials because they had not been licensed, and even empty oxygen generators were known to discharge toxic residue. Regardless, the ValuJet ramp agent accepted the shipment and together with Richard Hazen, the copilot, who also should have known better than to accept the oxygen generators, determined that the boxes of oxygen generators and three spare airplane tires would be placed in the forward cargo hold.

It is believed that the heat generated by inadvertently fired oxygen generators was more than adequate to catch the cardboard and spare tires on fire, and that the cabin was quickly filled with flames and toxic, black smoke. While the pilot and copilot should have had oxygen masks that would have protected them from the toxic smoke, they either did not get the masks on in time to avoid being poisoned by the smoke or the fire may have spread to the cockpit. Whatever the reason, the pilots were unable to turn the plane back towards Miami International Airport, from which it had taken off, and the plane went into a dive into the Florida Everglades, despite there being no damage to the outside of the plane.

A tragic accident had occurred that could neither be blamed on a failure of engineering nor blamed on a single obvious mistake. The ValuJet 592 crash was the result of several errors that culminated into a horrific accident and the loss of several lives.

David Hinson, then administrator of the FAA, came out and made an announcement that flying on ValuJet was safe and that consumers should not be concerned. However, within a few days it became known that inspectors with the FAA had been concerned about ValuJet because of a "disproportionate number of infractions" and because the company had been growing too quickly without the procedures or people to maintain the requisite level of safety. Upon review after the crash it became apparent that ValuJet may have warranted being shut down prior to the crash, and David Hinson drew criticism for not having taken action that may have prevented the accident. Five weeks after the accident ValuJet was permanently grounded.

The article notes that Charles Perrow blames this type of accident where failures occurred at many levels on a combination of "tight coupling" and "interactive complexity," by which he means that many elements that are involved are "linked in multiple and often unpredictable ways." This complexity in the relationship of these complex systems allows the failure of one part to "coincide with the failure of an entirely different part," which can then cascade to additional entirely different parts due to the "tight coupling." The complexity and rigidity of the airline industry had done its work on ValuJet 592, and the resulting "tangle of confusion" had led to the tragic accident.

Langewiesche notices that the work order from ValuJet called for "expired" oxygen generators to be removed, and that those generators which had not been "expended" should have a plastic cap placed on their firing pin. Langewiesche notices the ambiguity in differentiating between these two words, and makes an argument that this technical "engineerspeak," was partly to blame for the accident. In addition to the technical jargon, Langewiesche sarcastically notes that an alternate way that the mechanics could have known how to properly dispose of the oxygen generators is by opening the "huge MD-80 maintenance manual, to chapter 35-22-01," line "h," which contained equally ambiguous "engineerspeak." There was excessive documentation that contained technical jargon not easily understood by temporary mechanics.

But it didn't end with just massive amounts of technical documentation that needed to be waded through; the industry is inundated by large amounts of paperwork. Langewiesche notes that this paperwork may be necessary, but it breeds a false sense of security. The presence of documents with signatures asserting that all things were done properly is no replacement for the actual, proper

execution of safe work. The lull that came with an industry that thought all must be safe despite corners being cut, the lack of concern for employees who seemed to care about their employers as much as their employers cared about them, a regulatory agency that did little to regulate the industry it was charged to protect, and a long list of small errors that all aligned into the crash of ValuJet 592 were tragic failures, but provide lessons to learn for our future.

2 On the failure of ethics leading to ValuJet 592

Ethics in terms of engineering is a term that may mean different things to different people. Few would argue that when a deliberate action performed with malicious intent led to the loss of lives an unethical thing had been done. However, ethics becomes ambiguous when it is difficult to pinpoint a failure on a single, purposeful action. Ethics requires the presence of choices that can be judged as either right or wrong in view of some coherent philosophy, and in sensitive industries where lives may be put in danger, special attention should be paid to the actions of the individuals who make decisions that can impact the safety of other persons.

Ethics in engineering cannot take the route of mere empirical considerations as John Stuart Mill's utilitarianism would suggest, nor should it be left to be determined wholly by reason as Immanuel Kant may argue. Ethics in engineering exists to protect the consumers, the society at large, and the community of engineers themselves. Ethics in engineering serves a specific purpose to a large group of people, but circumventing ethical concerns may pose beneficial to some individuals and organizations. Because of the necessity of ethics in encouraging the broader society, ethics in the realm of engineering, here including the airline industry, should be a function of governing. Whether governed by an independent body or by the state government itself, ethics in engineering serves a specific function of government. Therefore, the ethics that were violated in causing the crash of ValuJet 592 do not necessarily have to stem from a singular school of philosophy, but may be relegated to a set of decisions made by the community in order to protect itself and its consumers.

One of the first "ethical" failures in this regard lies with the creation of ValuJet itself. As the company's name may imply, ValuJet existed to provide a cheaper solution to those wishing to fly, but it isn't possible to focus on cutting costs and provide the same attention to providing safety. Langewiesche notes that "safety is never first, and it never will be," because of the competitive nature of flying. It is expensive to fly, but there are increasing amounts of people who are willing to pay a reduced fare to fly. This offers a growing market to airlines who are able to find ways to save money, even if this means cutting corners. And that is exactly what ValuJet was doing; when ValuJet 592 took its fateful flight ValuJet was a rapidly expanding company that had found many ways to reduce costs by cutting corners.

The corners that were being cut started with its employees. One of the first things that many managers learn is that one of the most controllable expenses is payroll. It made sense to ValuJet to forgo permanent, full-time employees and use temporary workers. Full-time employees require benefits, require leaves of absences, are protected by laws, require payroll taxes, and have to be paid even when there is insufficient work to warrant their presence. The use of temporary employees allowed ValuJet to save a lot of money, regardless of what consequences that may bring. While paying more money to an employee does not necessitate better, safer work, it is no surprise that a corporate culture which paid little attention to its employees was doing little about numerous minor safety infractions that had been appearing. ValuJet's primary concern was not with empowering its employees to make safe actions but with saving money wherever money could be saved.

Another "ethical" failure existed at the lowest level, the failure of each individual worker who had contact with the oxygen generators which eventually landed in the forward cargo hold of ValuJet 592. While the company and the industry were guilty for the culture which fostered and ignored a lack of safe behavior from the individual employees, the employees themselves were willfully acting within this broken system. Multiple occurrences of "pencil-whipping" and the acceptance of hazardous materials by both a ValuJet ramp agent and copilot may have been considered a normal course of action to the individuals at the time, but it is clear in retrospect that these actions were not safe or appropriate. The nature of the system accident is such that no single individual can receive blame, but there are a series of individuals that can share it. If any one of the many individuals who had contact with the oxygen generators had thought to speak up and question the presence of those generators, the ValuJet 592 crash may never have happened. Langewiesche and Perrow, however, may argue that an individual speaking up would do nothing but delay such an accident.

Charles Perrow argues that contrary to Murphy's law, "what can go wrong usually goes right." Everyone had grown used to taking shortcuts, and as a result it was natural for everyone who came along and worked with the expired oxygen generators to ignore the safety threat that they posed. While an individual speaking up and stopping the oxygen generators that were destined for ValuJet 592 may have saved that flight, the broken system itself would have eventually fostered another similar incident just at another place and time. Because of the widespread nature of the problem, one cannot stop at recognizing the "ethical" failures of the individual and the company, but must go beyond to the broadest sense.

The crash of ValuJet 592 was a result of multiple failures, but it may perhaps stem from an "ethical" failure on the part of the airline industry as an aggregate whole. Langewiesche notes that following deregulation of the airline industry it made sense to find ways to save money, and that cutting corners had become a routine practice across the industry.

David Hinson, administrator of the FAA, following the crash, went to Congress and requested that the FAA's "dual mandate," which included promoting airlines, be removed. Langewiesche refers to this as mere theatre, but it is telling that the federal agency primarily in charge of the safe function of the airlines was initially charged with promoting the industry itself. This mandate to promote the industry seemed contradictory to the requirement of ensuring the safety of consumers. It translated into an agency that, along with the NTSB, had little actual ability to regulate anybody. The FAA did exist to promote safety, but with contradicting mandates and not enough people to watch over every airline employee, it was wholly inadequate to ensure anyone's safety. The best that the FAA could do was outline rules and standards for airlines to abide by, and these led to the large number of processes making up the broad, complex system that the airline industry had become.

The broad, complex airline industry laid down many procedures to abide by, but cared little about the actual result of the actions of the employees working for the airlines. Paperwork and precise instruction existed covering nearly every mistake that led to the crash of ValuJet 592, but no one in the industry cared to think beyond the proper execution of the paperwork and the saving of costs. The nature of the airline industry has become a mere process that has moved away from the very human dangers that are present in the improper execution of the functions of employees working in the industry.

The "ethical" failures of the airline industry, ValuJet, and the employees that handled the oxygen generators that ended up in the forward cargo hold of ValuJet 592 were many, and can be summed up in many numbers of ways. But the ultimate failure of the industry as a whole was a

lack of focus on actual safety and a focus on merely the processes laid down by the FAA and the complex system that had to be navigated in order to qualify as a "safe" airline and make money.

3 What Computer Software Engineering can learn from ValuJet 592

When planning, designing, and developing a software product or application, there are many things that can go wrong. There are even many software endeavors that bear consequence on people's lives. There are even some widely used industry standards that layer complex safety documentation and checks in place, just like the airline industry. While there are many differences between software engineering and the airline industry, there are still more than enough parallels to learn lessons from the system failure that happened with ValuJet 592. By taking these lessons to motivate the individual employees, remove false complacence from managers, and properly test software, it may be possible to mitigate the number of system errors that arise in software engineering.

The pilot, Captain Candalyn Kubeck with significant experience flying, earned about $43,000, and her co-pilot, Richard Hazen with similarly significant experience, earned about half of that. The article notes that along with the pilots, flight attendants, ramp agents, and mechanics were all paid much less than a more traditional airline would have paid. ValuJet became notorious within the industry for its use of temporary employees and independent contractor such that some began to call it a "virtual airline." All of this success, however, led to rapid expansion, and some FAA regulators had even begun to become concerned about whether ValuJet could really maintain quality and all the paperwork required at the rate that it was growing.

When the oxygen generators that may have caused the ValuJet 592 crash were removed from the three MD-80 aircraft that ValuJet had purchased, they were done so by a contractor that ValuJet had hired, SabreTech. SabreTech was a large firm that often did this type of work, but the article notes that SabreTech routinely hired contract mechanics that worked on an as-needed basis. This resulted in about three-fourths of the people on the ValuJet project being temporary workers that had been contracted. Not only was SabreTech using contracted temporary employees, but it also wasn't treating them very well. When the contract deadline was drawing close, SabreTech had employees working on shifts day and night and even on some weekends. SabreTech, like ValuJet, was finding ways to save money by cutting corners; minimizing payroll, benefits, taxes, and other expenses that come with hiring full-time employees; and not treating their employees well.

While neither ValuJet nor SabreTech seemed to care about their employees very much, it is not possible to argue that if either ValuJet or SabreTech paid their employees more this accident may have been avoided. That does not mean, however, that considering the way in which both companies saved money when it came to personnel should not be taken into consideration. The fact that both companies avoided full-time staff and worked with a large number of contracted temporary workers was a great way to save money, but it was not a great way to get employees who were willing to work very hard. Through several steps of the process by which the oxygen generators which may have driven ValuJet 592 into the ground ended up on the airplane, it was apparent that the employees had not taken too much consideration into their jobs; Langewiesche calls this "pencil-whipping." The bottom line was that the employees were not motivated to do anything beyond what they had to in order to receive their paycheck.

Dan Pink at a talk for the Royal Society for the encouragement of Arts, Manufactures and Commerce, the RSA, mentioned that one way to motivate people in jobs that are knowledge-based is to, "pay people enough so theyre not thinking about money but thinking about their work," and that the three factors that lead to better performance after money is taken "off the table" are

autonomy, mastery, and purpose [Pink, 2010]. His fundamental argument was that you could not have employees that were motivated to try, work hard, and care about the results of their work when their primary concerns lie apart from the actual job. Computer software engineers play an important role in software development and are empowered to do great jobs or make critical errors in the course of their software development. While it is not necessary to just pay each employee a lot of money, Pink's argument is that employees should not spend their time thinking about ways to make more money. Employees should spend time caring about the work and the results of the work that they are performing. It is not easy, however, to motivate this kind of conern.

Autonomy, mastery, and purpose are hard to foster, and are made even more difficult by the presence of layers of safety guards and standard procedures designed to remove autonomy. Computer software development cannot rely on a single planning stage in the development of software. The computer software engineers who are tasked with developing the mission critical software that lives depend upon will have little incentive to catch mistakes and do more than the minimum required of them to perform their task to completion if their only task is to meet the rigid requirements of an abstract document.

A perfect example of this was an anecdote related by Professor Robert "B.J." Johnson of Loyola Marymount University. B.J. related a past, large-scale software development project that he had participated in where his immediate supervisor was under pressure to meet time deadlines for his project. B.J. had created a Java class, a sub-program part of a larger program, that was not of major importance, but had not implemented any actual functionality in the class. When the supervisor came by and asked him how the class was doing he explained that the class existed and could successfully compile into a working subprogram, but did not have any functionality. The supervisor considered the importance of completing the project on time and instructed B.J. to place the class with no functionality into the repository of completed software classes and mark the class as done. This falsification of completion bears so much resemblance to the "pencil-whipping" that occurred with the ValuJet 592 project that it is hard not to draw the parallels.

The waterfall model of software development is a sequential software development model commonly used in highly sensitive projects where safety and security are highly important. This was modeled after many other engineering methodologies and relies on front-loaded customer involvement where requirements are clearly specified to great extent in extensive and complex requirements specifications. However, as the anecdote that Professor Johnson related exhibits, there are times when these detailed requirements specifications fail to translate into ideally executed software. The over-specification of the development process creates rigidity that leads to engineers who live with the mentality that it is more important to just get the job done than to do it well. The existence of detailed requirements documents lulls managers into a false sense of security and allows them to assume that all aspects of a software may work correctly if all elements of a requirement specification are checked off.

However, the simple removal of the initial software design process would not by its own right improve the quality of the code produced. Robert "Bob" Martin, in his book Clean Code, discusses the importance of customer, or manager, involvement throughout the development process. When there are actual stakeholders involved in the development stage of software, "pencil-whipping" is less likely to occur. While some may pose the argument that the resources required to actually have stakeholders review the software produced over short iterations and constantly review the specification may not be worth whatever benefit it may convey, Martin argues that companies cannot afford to not have stakeholders review throughout the development. Front-loading the design process turns the development stage into the exact mindless job that lacks any motivation

that the workers who had encountered the oxygen generators that were destined for ValuJet 592 had faced. The presence of rigid timelines and simple paperwork to ensure success has proven to be inadequate in ensuring proper execution. By acknowledging this failure, managers and stakeholders can take a more active role in development and not carry a false sense of complacency fostered by the presence of paperwork attesting to the proper engineering of software.

It is important, still, to take into consideration the contribution of every individual in the software development process. ValuJet and SabreTech proved that they did not care about their employees, and their employees proved that they did not care about their employers. The programmers who create the software outlined in design documents are the ones that are most likely to see, or even cause, the errors that can prove mission, or life, threatening. By fostering autonomy, mastery, and purpose, managers can empower programmers to think beyond the box of checking of paperwork, take ownership of the produced software, and act in ways to ensure the development of the best code possible.

New development models, such as Agile, depend on the programmers themselves to design the lower level implementations of some features. By giving ownership of some design aspects of the project to the team producing the actual software, it fosters autonomy and ownership in the programmers who are then more likely to be concerned about functionality that fails.

Despite all the safeguards that anyone might concoct for software engineering, there is always the chance of errors occurring that are unforseen and unexpected. The benefit that software development has over the airline industry is that through unit and integration testing many problems can be exposed before they ever reach a consumer who might be harmed by it. There are tools out today that measure code coverage by unit tests and even software disciplines that advocate the use of tests on every piece of code before the code is ever written. Test-Driven Development (TDD) advocates programmers to write unit tests before any code is ever written to ensure that every code that is ever written is always being tested. The proper coverage of unit and integration testing cannot guarantee the removal of complex system errors, but with high code coverage in tests it is possible to make it more difficult for mistakes to slip through the "system."

4 Conclusions

Langewiesche and Perrow both argue that "system" errors are born out of the complexity of the processes that we create, and cannot be entirely avoided despite what safeguards one may try to put in place. Software engineering contains some parallels to the airline industry by which lessons may be learned from the crash of ValuJet 592. These lessons may be learned, but they still may not completely remove the existence of errors.

Computer software engineering could try to mitigate the occurrence of "system" errors in the future by addressing the problems that the airline industry did not address in the days leading up to the crash of ValuJet 592. The bottom-line, however, is that ValuJet 592 crashed because people didn't care. The airlines had more incentive to care about profits than about safety, the employees cared more about appearing to get work done and keeping their jobs than about doing a job well, and regulatory boards did not have much of a chance to be concerned about anything at all. The suggestions we considered in the previous section are all geared towards the hope that it is possible to make more people care about the process.

By involving stakeholder in software design throughout the development process, we force constant review and consideration by the stakeholders. By giving autonomy and purpose to the programmers by giving them a sense of ownership through a share of the design process, we encourage

the programmers to care more about the software they are producing. All the documentation and procedures that exist may offer a hollow sense of accomplishment to those charged with the safety and security of the software produced by the software development engineers, but if we cannot find a way to make more of the actors within the "system" to care about the results of the process it really may be impossible to avoid these "system" errors.

Fortunately, one way in which computer software development differs drastically from the airline industry is the strength of its ability to test for errors before deploying its products. Through simulation, integration testing, and unit testing, it is possible to catch many errors before they cause harm to anyone. By properly testing each piece of software and empowering those charged with the testing, errors can be made less likely. However, without addressing the nature of these complicated "systems" where the individuals inside them care little about what happens as a result, the people testing the software are little more than an additional cog in the broken wheel. If we were to couple these test processes, however, with the motivation that can be endowed to the software developers and stakeholders, then it may very well be possible to avoid "system" errors for a little while longer.

References

[Langewiesche, 1998] Langewiesche, W. (1998). The lessons of valuejet 592. *The Atlantic Monthly*, 281(3):81–97.

[Pink, 2010] Pink, D. (2010). Rsa animate - drive: The surprising truth about what motivates us. `http://www.youtube.com/watch?v=u6XAPnuFjJc`.