ECEN 5623 – Real Time Embedded Systems

Summer 2017

EXERCISE 5

Studying Popular Case Studies of System Failures and Project Proposal

Jeet Baru

Email: jeet.baru@colorado.edu

ID Number: 107189037

Question 1

In this question we are supposed to research software and hardware computer engineering design flaws of all time. I studied Blackberry Storm, Windows Genuine Advantage, Windows 8, Windows ME, Apple Lisa, Pentium FPU Bug, Norad False Alarm, Mars Express Beagle 2 and Challenger and Columbia Shuttle Loss. On reading the above case studies, I have listed the following to the 3 worst in terms of negative impact, negligence and bad decisions made. I have listed them in the order of worst to least bad.

Challenger and Columbia space shuttle disaster:

The Challenger space shuttle, launched by NASA disintegrated within 73 seconds of the launch due to failure of the right Solid Rocket Booster – leading to the demise of 7 astronauts on-board. This is one of the worst mission failures primarily due to the cost of human lives involved in it. The Challenger shuttle was lost due to an explosion of the external fuel tank when a rocket booster came loose and ruptured the tank. Essentially, a seal around the primary O-ring (which is a component of the rocket booster) failed leading to hot gasses leaking out of the rocket boosters which eventually burnt a whole into the fuel tank – and lead to an explosion. In my opinion, the biggest cause of this failure was insufficient testing of all parts which is especially necessary in such a critical mission involving humans. It turns out that the O-ring had not been sufficiently tested at low temperatures. At low temperatures, the O-ring hardens and is not able to effectively form a seal – creating gaps through which hot gasses could escape. Also, there was cold weather on the morning of the launch, which aggravated the problem. Although the problem for this failure was physical but sufficient testing and negligence would have prevented it. The price to be paid was a failed mission and the loss of seven lives.

The Columbia Space shuttle disaster was other such catastrophe. There were 7 astronauts on-board and on the shuttle's re-entry into Earth's atmosphere, it broke down resulting in the demise of the astronauts. Essentially, the presence of hot gasses in the left wing caused the disintegration of the wing on re-entry and as a result the orbiter broke. The hot gasses inside left wing were primarily due to two reasons – presence of hole in wing due to foam falling off of an external tank and striking the wing during ascent and the second reason being the shuttle traveling at high velocities during re-entry. The falling off of foam had taken place in many of the past missions as well leading NASA to consider it as a low-risk. The ground control engineers had made 3 separate requests for high-resolution images taken by the DoD which would provide an estimate of the damage and also NASA's chief thermal protection system engineer requested that astronauts on-board be allowed to space-walk to inspect the damage and potentially repair the problem. However, the management at NASA turned down these requests. This incident and the Challenger shuttle crash incident prove that thorough testing of all components involved in such a critical mission is absolutely essential. One must not be complacent based on past successes.

In my opinion, the failure of the Challenger and Columbia Space Shuttle missions was the worst. This is because the cost in these missions was not only a failed mission (in the case of Challenger) after putting in enormous time, money and resources but also the loss of human lives.

Windows Genuine Advantage:

Windows Genuine Advantage is essentially an anti-infringement system created by Microsoft to enforce online validation and authenticity of the installed Microsoft Windows OS when accessing services like Windows Update and downloading components from the Download Centre. WGA consists of an installable component that hooks into Winlogon to validate the Windows license on each logon and also an ActiveX control which checks validity of Windows license when downloading updates from the Download Centre. One of the major accusations levelled against Microsoft were that they were using WGA for spyware like behavior, by "phoning-on" the user and collecting information about the user without the user's consent. Another issue was that of false positives. Around 20% of the computers running Windows failed the test even though they had genuine Windows software installed. The company was criticized heavily for this. Apart from that, it also gave false negatives as Windows coulde be validated using IEs4Linux and Wine running on Linux. The wrongly identified people (false positives) are subjected to repeated and annoying messages that their copy of Windows is fake and they should take corrective action. I decided to include this in my list of worst softwares/hardware due to the negative impact that one software like WGA had on the reputation of Microsoft.

The failure of Windows Genuine Advantage and the hit that Microsoft makes it the second worst failure in my opinion. There were allegations of it being a spyware that were levied and also Microsoft was forced to accept that it was "phoning home" every day and would be reducing its frequency to once in two weeks. Many customers who had a genuine copy of Windows installed were hassled by this software that they had a counterfeit version (false positive). All of this resulted in Microsoft's goodwill taking a hit and its software being marked as "neutral" down from "good".

Apple Lisa:

Apple Lisa was one of the first desktop computers developed by Apple to provide a GUI aimed at individual business users. It was released at a very high price of $9995 with a limited amount of memory of 5 MB in hard drive usage. The high price combined with low performance and unreliable floppy disks resulted in poor sales for the product and Lisa turned out to be a big failure. Lisa was unique in many ways in that it was the first product that had a graphical OS in it and it also employed non-preemptive multitasking and virtual memory. The market had cheaper alternatives in IBM machines. Also Steve Jobs had announced that they will be releasing a superior system which would not be backward compatible, this too prevented people from investing in the machine. The intended business customers were reluctant to purchase it because of its high price and the result was a big commercial failure and Jobs getting removed from the team. Hence, the failure was not due to any real-time failures or bad interactive system design but purely due to bad decisions regarding the pricing.

The failure of Apple Lisa able to dominate the desktop market and generate the profits that Apple expected it to generate make it the third worst failure in recent history. Lisa had almost everything right for its time – non-preemptive multi-tasking, virtual memory and a graphical OS. However, the one thing it did wrong – its exorbitant price resulted in meager 10000 units getting sold.

Question 2

I read case studies of real time mission critical design errors of time including: Three Mile Island, Mars Observer, Ariane – 501, Cluster spacecraft, Mars orbiter, ATT 4ESS upgrade, Therac-25, Toota ABS software. I have identified the following 3 to be worst of them all. They are listed in the order worst to least bad.

Threac-25 disaster:

The Therac-25 was a radiation therapy machine produced by Atomic Energy of Canada Limited (AECL) in 1982 after the earlier Therac-6 and Therac-20 units. From 1985 to 1987, it was involved in at least six accidents between 1985 and 1987, in which patients were given massive overdoses of radiation. Because of concurrent programming errors, it sometimes gave its patients radiation doses that were hundreds of times greater than normal, resulting in death or serious injury. The reason behind this accident was due to the activation of high power electron beam instead of the intended low power beam. This happened due to concurrent programming errors which was ultimately responsible for radiation overdoses. Earlier models had hardware interlocks in place to remove these errors, but Therac-25 had replaced these with software interlocks instead for safety purposes. Software interlocks failed due to race condition i.e. frequent overflowing of one-byte counters in a testing routine. If an operator provided manual input to the machine at the precise moment that this counter overflowed, the interlock would fail. The root cause of this issue can be traced to not having the software code independently reviewed. Failure modes due to mal operation was not identified. The issue occurred only when there was an improbable inputs via keystrokes. Also on occurrence of the error only the word MALFUNCTION was displayed. The user manual did not explain error codes. Rather, being a critical error the system should have shut down. Moreover, the initial reports of issue were not believed due to overconfidence of personnel.

In my opinion, the failure of Therac-25 was the worst. This is because the cost was not only money but also the loss of human lives. Due to programming errors, patients' radiation doses were hundred times than normal which caused several deaths and serious, which makes it the worst real-time design and implementation failures.

Toyota ABS Software:

In 2010, Toyota had to recall approximately 400,000 Toyota Prius Hybrid cars in order to update the software controlling Anti-Braking Systems (ABS) of these cars ultimately improving the brake response. Problems regarding the braking systems of these cars were reported worldwide by around 102 drivers, especially on slippery surfaces like ice. Car accidents were reported injuring people. The reason for failure was the delay in regenerative braking when hitting the bump, leading to long stopping distances. These problems occurred because the braking systems failed to meet the deadlines which resulted in longer braking times. Toyota corrected this mistake by re-installing the new software in the old cars as well as, by default, in new cars. A US federal grand jury in New York began the process of determining if they had to charge Toyota criminally for the way it has handled the Prius' brake recall, and a civil class action lawsuit was filed on behalf of 2010 Prius hybrid owners ultimately costing them monetary losses.

The failure of Toyota's ABS software, was potentially a life-threatening failure and if not for the complaints the drivers worldwide, it could have resulted in numerous serious accidents. Also, it cost

Toyota, the extra time and resources wasted on updating the software on old cars. Hence I feel it the second worst error among others.

Cluster spacecraft failure:

Cluster was a constellation of four European Space Agency spacecraft which were launched on the maiden flight of the Ariane 5 rocket, Flight 501, and subsequently lost when that rocket failed to achieve orbit. In 1996, the launch ended in a failure due to an error in software design caused due to assertions being turned off. Due to this, there was inadequate protection from integer overflow. After 37 seconds, the rocket started veering from its path, then it started to disintegrate due to aerodynamic forces and due to its automated flight termination system, it ultimately got self-destructed. Due to this software error, it resulted in a loss of more than US$370 million. The greater horizontal acceleration in Ariane 5 caused a data conversion from a 64-bit floating point number to a 16-bit signed integer value to overflow and cause a hardware exception. Due to efficiency considerations, they decided to not do range checks on this variable even though conversions of other variables in the code were protected. The limitations of the alignment software were not fully analyzed and the possible implications of allowing it to continue to function during flight were not realised. The specification of the inertial reference system and the tests performed at equipment level did not specifically include the Ariane 5 trajectory data. Consequently, the realignment function was not tested under simulated Ariane 5 flight conditions, and the design error was not discovered. It was decided to use the simulated output of the inertial reference system, not the real system or its detailed simulation. Had the system been included, the failure could have been detected. The launch failure brought the high risks associated with complex computing systems to the attention of the public, politicians, and executives, and due to this support for research on ensuring the reliability of safety-critical systems has been increased.

According to my opinion, the failed mission of the Cluster spacecraft launched on the maiden flight of the Ariane 5 rocket, Flight 501 was the third worst real-time mission critical failures of all time. This is because the cost in these missions was not only a failed mission after putting in enormous time, but it costed them a loss of US $370 million making it one of the most infamous and expensive software errors in history.

Question 3

The USS Yorktown reported to have a real time interactive system failure after an upgrade to use distributed Windows NT mission operations support system. We read papers related to an incident that left the USS Yorktown disabled at sea.

Summary of Gregory Slabodkin's Findings:

Microsoft Windows NT was installed on the ship so that fewer sailors would be needed to control the ship. This would save navy lot of money. For his report in GCN, Gregory Slabodkin mentions that the Aegis missile cruiser USS Yorktown has suffered multiple software glitches resulting in system failures. He focuses on a specific incident in which the system failure caused the ship to be towed to the Naval base in Norfolk, VA.

The Yorktown suffered system failure when bad data was fed into the computers during manoeuvres off the coast of VA. A civilian engineer, Anthony DiGiorgio, mentioned that a database overflow caused the propulsion system to fail and it required 2 days of pier side maintenance to fix the problem. A memo later released mentioned that a system admin entered zero in a data field of the Remote Database Manager. This resulted in a divide by zero glitch which caused the database to overflow and crash all the LAN consoles and remote terminal units of the propulsion system. The memo also mentioned that it was the second time there had been an issue with the control system and the first time involving the propulsion system. The admins are trained to bypass a bad data field in case such a problem occurs again.

DiGiorgio says that Yorktown's failure was not as simple as reported. He mentions that a $2.95 calculator does not crash when we try to divide by zero, maybe the Yorktown wasn't designed to tolerate such a simple failure. He further criticizes windows NT to be the root cause of the system failures. Unix is a better OS for control of equipment, whereas NT is better for data transfer. There have been multiple shutdowns caused by NT and USS Yorktown has been towed into the port several times for the system failures. NT based LANs experience a domino effect where the failure of a single computer leads to the failure of the entire system.

Other article in contrast to the previous report:

'Smart Ship inquiry a go' https://gcn.com/articles/1998/08/31/smart-ship-inquiry-a-go.aspx. This is another article by Gregory Slabodkin posted nearly a year after the incident. This article mentions the Navy Chief Information Officer, Ann Miller, saying that the Yorktown incident has no issue with NT, rather it was a basic programming problem. The Navy has continued faith in Windows products and all new apps must run under NT. Windows officials said that NT did not cause any issue as production level data-filed safeguards were not installed by the software programmers in the Yorktown program. The articles highlights that the root cause for the failure was an erroneous entry into the software, which should have been handled by the software engineering staff hence lifting the blame from Microsoft.

Hence we see two articles from the same author providing 2 contrasting opinions of the root cause of the issue.

Root Cause of the Issue:

I feel the root cause for the incident was to allow user to manipulate and enter data that could cause system wide failure. The user should not be allowed to modify system core memory, at least not without any warnings. The operator interface is also to blame for this as the interface allowed the modification of the data field. The interface should not have exposed system core memory and also if it is required, no modifications should be allowed before having root privileges.

I also feel that proper error detection and handling was not incorporated in the software. Also the domino effect of failures could have been prevented if proper error detection was implemented.

Why and why not use Linux over windows for this application:

Why Linux:

- Linux has a user space and a kernel space and thus user applications can safely run in the user space without making any system critical kernel changes.
- Linux shall be better system for Control and Machinery and while Windows NT used for data transfer.
- Linux according to me is more reliable with higher uptime.

Why windows:

- As mentioned earlier Windows prove to better in applications requiring high data transfer needs.
- Windows shall provide for a better and user friendly interface compared to Linux.

Question 4 – Project Proposal

I shall be doing the extended lab "Timelapse" as my final project. I shall be doing this project alone. I shall be using a BeagleBone Black as my target device and shall be using Logitech's C200 camera.

I have listed the requirements and the dates by which I plan to them.

Minimum Requirements:

- The camera being used (Logitech C200) shall provide me with the required resolution of 640 x 480.
- With my experience of using the hardware in the previous lab I have observed that the camera supports ppm format encoding.
- To achieve the requirement of 1Hz for timelapse, I plan on acquiring and timestamping frames on a continuous rate and select only those that have a consecutive time difference of 1 sec. I am not sure if this shall be a correct approach as I am not sure of the timing requirements of embedding the timestamp in the PPM header of each frame.
- **Also my observation was that the C200 camera gave me only 15-17 Hz of frame rate as compared to the minimum requirement of 24 Hz. Will this hamper my ability of meeting the deadline for timelapse?**
- My approach to the problem shall be implementing an algorithmic solution on my host machine and check for results by August 5th and implement and check ensure timing closure by August 7th on the BeagleBone Black.

Target Requirements:

- I plan on implementing 2 of the target requirements, including compression of frames so that I am able to save larger amount of frames.
- I also plan to implement either transfer of compressed image over Ethernet or try saving it onto inserted SD card. I expect the time required to save the image externally to be significantly greater and posing a challenge to my codes ability to meet the deadline.
- I plan to work on Target requirements by 10th August.

Stretch Goal and verification:

- I wish to implement 1 stretch goal. I can try and trigger timelapse to begin when I can observe change in RGB pixel value summation thereby trying to achieve the same goal but now by detecting physical change in the clock face.
- I also wish to spend the remaining time on verification requirements by ensuring accurately timed image acquisition and observation of physical process at the same time.
- I plan to work on the report along with my implementation and thus plan to complete the project by 12th August.