Class 304: Fantastic Failures

Kim Fowler

The Comet

On 2 May 1953, a De Havilland Comet jet airliner took off from Calcutta, India and disintegrated in mid air. The disaster was blamed on a tropical thunderstorm. The following year, two more Comets exploded: one in January 1954 and the second in April. Both airliners had taken off from Rome and were climbing to high altitude. The January accident happened in mild and clear weather. [1]

The Comet was the world's first commercial jet airliner. Developed in Britain in the late 1940s and early 1950s, it had a pressurized cabin and flew at high altitudes above turbulent weather. The Comet was the pinnacle of aviation for speed and comfort. The designers claimed that the airliner had "flown off the drawing board" because it went straight into production without a prototype. [2]

In the investigation that followed, it was learned that the forward parts of the airliner had broken apart before the tail. The extensive testing involved immersing an entire fuselage in a water pressure tank and cycling the pressure to emulate the changes in pressure during flying. The engineers found catastrophic cracking from metal fatigue associated with high stresses near the corners of windows and fuselage penetrations. The cracks rapidly, and explosively, widened to destroy the airliner.

The fixes included rounded corners on windows, stress-fracture resistant plates around fuselage penetrations, and new ways of fabricating high-pressure fuselages. The general understanding of metal fatigue and its effects also increased through vigorous attention from metallurgists, aircraft designers, and researchers.

What can we learn from such events? The Court of Inquiry reported, "Throughout the design, de Havillands relied on well-established methods essentially the same as those in general use by aircraft designers. . . . They believed, and this belief was shared by the Air Registration Board and other expert opinions, that a cabin that would survive undamaged a test to double its working pressure . . . would not fail in service under the action of fatigue. . . ." Petroski writes, "Their belief was of course wrong, and the experience with the Comets eventually improved the state of the art of aircraft design" [3].

Why study failures?

Failure can be the source of useful information; it can advance the state of the art. Petroski has written, "No one wants to learn by mistakes, but we cannot learn enough from successes to go beyond the state of the art." [4] With the Comet, engineers pushed the envelope in aircraft design; they gained a better understanding of airline design after the failures.

You cannot possibly predict all circumstances or failures that might occur. Studying failures helps you to understand the sources of failure and extenuating circumstances. From these you learn to constrain the unknowns, which could lead to failure, in your own system designs.

The nature of failure

Most failures are not single-point; generally a single event does not entirely account for the failure. Often multiple problems interact to precipitate the failure. Multiple initiating events and unforeseen circumstances usually combine to force the system into regions of complexity that the human mind cannot easily predict or even comprehend.

Nonobvious use and improper operation can lead to failure. I have heard that half of all hospital accidents are due to improper use of correctly operating equipment. [6] This places the burden on you, the designer,

to develop intuitive, useful, and obvious interfaces. Another general characteristic of failure is that it usually takes a while for the serious problems to appear. They don't happen frequently, immediately, or predictably.

Example: NIST Encounters Hurricane Isabel
In September 2003, Hurricane Isabel caused power outages in the Washington, D.C. area because the wind blew trees down onto power lines. The National Institute of Standards and Technology (NIST) did not have a power outage but did experience 180 VAC over-voltage for 20 minutes that destroyed 1000s of fluorescent lamp ballasts. The protective mechanisms for AC power, which could have prevented the damage, were controlled over telephone lines. Guess what was also knocked down by falling trees? [5]

Clearly, the designers did not realize that power problems and the loss of protective control could have the same source. They also may not have realized that power outages in the region would cause extended periods of over-voltage that could damage installed equipment.

Factors that set up failure

A major factor is lack of foresight and leadership from management. Schedule pressures and low estimates always "scrunch" review and testing the most. Why is it that we never seem to learn that careful review and thoughtful testing are crucial to building products? Appropriate testing can reduce unpredictability and reveal problems with multiple factors and improper or nonobvious uses. Some factors that cause schedule slip are underestimation of effort, changes in project scope, and outside pressures. [7]

Too many features, i.e. creeping featurism, makes interactions within designs difficult to comprehend. Sometimes when we "reinvent the wheel" and try to do it just a little better than what we can buy, we set the stage for reinventing and solving all the problems – again.

Perceptual failure is just as bad as a catastrophic failure - people won't use your product. What users perceive about your instrument is their reality of its value. It may function correctly but if users don't like its looks, they will find reasons to avoid using it. So, if you got the technical operation right, but the basic look-and-feel isn't acceptable to users, then you have a real, big problem. Let's be blunt – it is a failure.

Preventing failure

You can do a number of things to reduce failure. Taken together, these activities will not guarantee success but they will help you avoid failure.

Build a boilerplate template for estimating schedules for project development. Use a basic spreadsheet or project scheduling software and calculate time and effort. Budget time and money for contingencies, even consider contingency plans for recalling products. Adjust and update the template after every project according to your newfound experience.

Plan to test. Leave plenty of time to test. Put it into a test plan. Real, live users should test your design. Engineers and people within your company aren't good enough; they are too close to the trees to see the forest.

Don't reinvent the wheel. Buy the real-time operating system, don't write it yourself. Buy the single-board computer; don't design one a little faster or a little fancier.

Learn from others. Read books, magazines, and newsletters. Clip and file articles. Attend conferences and trade shows. I have several favorite columnists and regularly follow their musings. Learn from your own

experiences; debrief and prepare a "lessons learned" document. Facilitate the increase of corporate knowledge within your company.

Teamwork knits together the organization and its development effort. Honest peer review and code inspections will dramatically improve the quality of your designs. You will also learn from your colleagues. "Iron sharpens iron, so one man sharpens another."[Proverbs 27:17]

Integrity

The definition of integrity is a whole, seamless entity. Integrity is all about the big picture. It is a clear, complete, and truthful perspective that provides a measure of assurance and confidence.

Why should you care about integrity? Because your reputation as a designer of instrumentation is at stake. You must know what you can do and what you cannot. You must be truthful in advertising your capabilities to your boss, your customer, and your colleagues. If you aren't truthful, projects cannot be properly planned and executed. Jack Ganssle bluntly puts schedule slips to, "We lie!"[7]

Do you take time early in a project to think through the big picture? Do you carefully set basic boundaries around the problem and its solution – simple calculations like voltage times current to determine power dissipation? Do you follow a development plan? Do you document thoroughly?

Estimating time and effort are matters of integrity. Make a science of estimation and planning. Study other people's experiences. Prepare development plans and checklists – and then follow them. Plan extra time for contingencies – particularly if this is the first time that you have designed this kind of instrument. One rule of thumb is to make your best estimation of effort and time and then double it. Much of my own work has followed this rule. Collect data and carefully record your experiences as they happen. Hmm . . . sounds like an engineering notebook! Use your experiences to refine your skills for estimating future projects.

Types of personal failure

Personal failure can occur in three different arenas: technical, professional, and political/societal.

You have the most control over your technical capability. Constantly seek to refine, mature, and build your technical capability from your experiences and from learning from others.

The professional arena involves reputations and relationships. You have less control over these since people, with all their quirks and flaws, are part of the equation. These people include your peers, bosses, and subordinates. In the long term, you can best build your reputation by basing your professional relationships upon integrity, willingness to communicate, and good technical skills. Still, there are no guarantees. Your career will take some unexpected and interesting turns.

Finally, you have the least control over the political arena where you live and work. Politics can reverse fortunes and project plans in an instant - the less people (who are in high places) know and the less time they have for consideration insures this kind of upheaval. You can influence it by being truthful. Eventually, the truth always surfaces but it may be after your lifetime. (A potentially bleak picture, but stick with me; there's hope yet.)

Personal failure has at least four sources: ignorance, stupidity, bigotry, and malice. Ignorance is the simple lack of knowledge, whereas stupidity is the refusal to use commonly available knowledge. You have the most control over ignorance and stupidity; learning and research help bound ignorance, while selective association in your relationships can avoid stupidity (i.e. stay away from stupid people – change

jobs if you must, it's your career and stupid people can ruin it). The remaining two, bigotry and malice, are more subjective and far more difficult to handle; they tend to arise from other people who are outside your control. Unfortunately, all four sources can feed on each other to intensify a bad situation.

To handle failure, persist in doing well. Accepting personal responsibility and doggedly pursuing excellence will help redeem failures for you. Admitting your own failures and supporting others who experience setbacks will keep you humble. Doing so will help you gain a clearer perspective on engineering and life.

Case Study 2: Ariane 5

The French have developed a family of launch vehicles for satellites. The previous generation of boosters, called the Ariane 4, successfully lofted a number of satellites. For a decade during the 1980s and 1990s, the French spent $8 billion to develop a new, larger booster called the Ariane 5. [8 – 10]

On 4 June 1996, the first launch of an Ariane 5 self-destructed within 40 seconds and lost four satellites. A shutdown of the control guidance allowed aerodynamic loads to separate the booster stages, which initiated the self-destruct sequence. Redundant computers in its inertial reference system (IRS) comprised the control guidance. Both computers in the IRS were identical and ran identical software. The IRS had been developed for the Ariane 4; it was reused in the Ariane 5 because of its successful operation with the Ariane 4 booster.

What went wrong?

A numerical overflow in an unused and unneeded software routine had shutdown both computers in the IRS. The design of the IRS would shutdown the computer upon numerical overflow. The shutdown occurred because three variables were left unprotected from calculation overflow in the floating point-to-integer conversions. The developers assumed they had a large enough operational margin to avoid overflow.

The IRS was developed for the Ariane 4 and fit the smaller booster's flight dynamics and performance. Unfortunately, the Ariane 5 had larger than expected drift in the horizontal velocity, which used up the operational margin and, of course, caused a numerical overflow. Since both computers ran the same software, when one experienced the numerical overflow and shutdown, the other – in lockstep – followed suit.

Simulation, ground tests, and extensive technical reviews did not detect the potential failure in the Ariane 5. No one performed any overall system tests to test all systems within the booster at the same time. A report from the investigation stated, "Limitations of software were not fully analyzed . . ." [11]

What we can learn the Ariane 5

Scaling up a design is a difficult task. Reusing a successful component in a different design does not guarantee success. This is where an understanding of systems and component interactions becomes important. Just because a component works correctly in one environment with its attendant set of interactions does not mean that you can infer that it will operate correctly in another environment. You must use both complete peer reviews and full systems tests, with functional production modules, to understand all the system dynamics and performance.

Case Study 3: Therac-25

The Therac-25 was a medical linear accelerator developed during the 1970s and early 1980s to treat tumors with high-energy beams. Between 1985 and 1987, Therac-25 accelerators had several accidents that seriously, or fatally, overdosed patients. [12]

Six patients accidentally received massive overdoses of radiation from the Therac-25 that resulted in serious injuries and death. In each of these cases, the Therac-25 delivered up to 250 Gy of radiation yet registered "no dose given." (5 Gy of whole-body radiation can kill.)

What went wrong?

The problems arose from a custom operating system that had a real-time executive using a preemptive scheduler. Systems analysis showed that dynamic issues in the multitasking environment led to the catastrophic overdoses. While each task ran well – and without failure – in isolation, asynchronous operation and communicating in the multitasking environment set up conditions for system failure.

First, a race condition caused by an operator's quick editing at the keyboard passed incorrect data to various tasks. Then a cryptic error message appeared with no follow-up explanation in the user manual, so operators did not know what remedial action to take. The Therac-25 depended upon software safety mechanisms; it had no adequate mechanical interlocks. For instance, it had a 1-byte status register that rolled over every 256 passes; a zero indicated that it was okay to proceed when that was not necessarily the case. Furthermore, the system did not have an independent physical limiter to prevent overdoses of radiation.

Follow-up investigations revealed that a single developer wrote the code without any checks, audits, or inspections. Very little of the software was documented. The development for the Therac-25 had no general plan and, consequently, there were no analyses for failure modes or hazards. There was no encompassing system test that exercised the entire system for dynamic interactions.

What we can learn from the Therac 25

Only good design produces a safe system; testing for faults does not guarantee safety. Leveson and Turner wrote, "Focusing on particular software bugs is not the way to make a safe system. Virtually all complex software can be made to behave in an unexpected fashion under certain conditions . . . The particular coding error is not as important as the general unsafe design of the software overall." [13] The unforeseen consequences of complex interactions often cause the problems. Even fully tested modules that exhibit no erroneous behavior themselves can produce wildly variant results when interacting with other modules and systems.

Again, this is a case where a systems view of component interactions and system synthesis is important. Understanding component interactions and nonobvious uses by operators is important in developing systems. This means that safety-critical systems need a general plan for system development that includes the full gamut of analyses, system design, design reviews, code inspections, tests, and system tests. Furthermore, independent systems should monitor and limit critical operations; an example would be hardware safeguards for software operations.

Case Study 4: Chernobyl

At 1:24 a.m. on 26 April, 1986, Chernobyl Reactor 4, a nuclear reactor, exploded in the Ukraine. During the following ten days, the reactor released clouds of deadly radioactive material into the atmosphere. People, who lived near the plant, received doses of radioactivity 100 times greater than generated by the Hiroshima bomb. The radioactive clouds drifted northward over Northern Europe and eventually around the world. The full consequences of this nuclear accident are still being studied. [14]

Russian engineers and technicians had been running an experiment on the reactor. A series of operator actions had confounding interactions that lead to the reactor overheating, which caused a meltdown of the core. Two explosions followed that blew the top off the reactor and the roof off the building.

What went wrong?

Soviet technicians prepared to perform an experiment with the nuclear reactor that powered a steam turbine. The experiment commenced around 1:00 p.m. on April 25. At 2:00 p.m., operators deactivated the emergency cooling system, but the reactor remained on the grid to supply power for unexpected demand. They took the reactor off line at 11:10 p.m. and planned to bring power output down to 25% of capacity. An operator shut down the automatic control and reduced power manually. The reactor self-damped, and by 12:30 a.m. its power output had dropped to 1% of capacity.

Operation at low capacity is dangerous in graphite core reactors of the Chernobyl type because they are unstable. Local bursts of reactivity can trigger a sudden start-up of fission throughout the reactor. Consequently, the operating rule for these reactors is to never bring them below 20% power capacity.

Understanding the immediate situation of low power output and the potential for unstable activity, the operators withdrew most of the control rods from the core to speed up reactivity and increase the power output. Within half an hour, power output had crept up to 7% capacity. The operators, under pressure to complete a test demanded by engineers in Moscow, which they perceived as a nuisance, then decided to continue the experiment; this proved fatal.

At 1:03 a.m., the operators turned on two pumps to circulate water through the core. This action brought the total number of operating pumps to eight and caused greater water flow through the reactor core that cooled it. Consequently, the steam pressure dropped because of the cooling effect from the extra water. Moreover, the operators apparently forgot that this extra cooling would cause an automatic withdrawal of more control rods from the core.

The experiment required the operation of the steam turbine, so the operators tried to increase the steam pressure by tripling the flow volume of the water. This unfortunate action did not produce more steam; it actually lowered its production and thus compounded the problem. With greater water flow, the reactivity of the core dropped, so the operators responded by removing even more control rods.

Steam production then rose in response, and by 1:22 a.m. the operators reduced the water flow to the reactor because they were satisfied with the production of steam. Now the reduced volume of water reaching the core was warmer, and steam production and reactivity both continued to increase.

At 1:23 a.m., as part of the experiment, the operators shut off the steam to the turbine. One minute later, they noticed something wrong and tried to shove the control rods back into the core to slow the reaction. The inordinate amount of heat in the core had bent the metal sleeves for the rods and prevented the rods from sliding into the core. The reactivity went unstable, and the water in the core flashed to superheated, highly pressurized steam. The reactor then exploded.

What we can learn from Chernobyl

Why did the operators continue the experiment at 1:00 a.m.? They knew about the instability problem but could not realistically conceptualize it. “Theoretical knowledge is not the same thing as hands-on knowledge.” [15]

Human nature tends to “over steer” dynamic systems; in operations, we view the immediate situation and not the differential. “We regulate the *situation* and not the *process*, with the result that the inherent behavior of the system, and our attempts at steering it, combine to carry it beyond the desired mark.” [16] This is called overshoot in control theory.

A compounding problem is that people cannot handle interacting, nonlinear problems. Dorner writes, “We find a tendency, under time pressure, to apply overdoses of established measures. We find an

inability to think in terms of nonlinear networks of causation rather than of chains of causation - an inability, that is, to properly assess the side effects and repercussions of one's behavior. We find an inadequate understanding of exponential development, an inability to see that a process that develops exponentially will, once it has begun, race to its conclusion with incredible speed. These are mistakes of cognition." [17]

Another confounding factor of human nature is to "groupthink" and to proceed arrogantly without independently assessing the possible consequences. Dorner writes, "The tendency of a group of experts to reinforce one another's conviction that they are doing everything right, the tendency to let pressure to conform suppress self-criticism within the group - this is what Irving Janis identified as the great danger of 'groupthink' . . . Everything the operators did, they did consciously and, apparently, with the complete conviction that they were acting properly. They did, of course, ignore the safety rules. But in doing so they did not neglect anything or do anything accidentally. Rather, they were of the opinion that the safety rules were designed much too narrowly for an experienced team." [18]

Shades of this type of expert reinforcement can be seen in the certification of the Comet's airworthiness by the design team, and the British government, before the problem of metal fatigue was more fully understood.

So how does this apply to embedded design? You should realize that complex interactions require clarity of function, and consequences, to reduce confounding problems. You should understand human nature, and design your systems to accommodate it.

Finally, we all need to learn to back up theory with actual experience. This means using thoughtful experimentation to advance theory.

Case Study 5: Apple Lisa

In the early 1980s, the Apple Lisa was a computer before its time. It embodied brilliant concepts for the human-computer interface by introducing the mouse and graphical file management to the commercial world. It was costly, on the order of $10,000 US dollars, and failed miserably as a commercial product.

A combination of factors conspired to prevent the Lisa's success. People were generally not ready for a paradigm shift in computer operation. They did not understand the utility of the Lisa. They were still using Apple Is and IIs, buying 8-bit Osbornes, and envying anyone who had an IBM XT with 20 Mbytes of hard disk. The software was either CPM or MS-DOS running single applications that had the basic look-and-feel of a terminal. The order of the day was to memorize hundreds of control sequences for keystrokes to do your word processing. Expert status accompanied those who used computers.

Certainly the price was prohibitive for an instrument that they did not appreciate. Even though the Lisa was too advanced, it prepared the way for the successful Apple Mcintosh a few years later.

What we can learn from the Apple Lisa

Great technological solutions must meet the perceived need. The utility of Lisa's mouse and graphical file management just wasn't appreciated at the time. Remember, only gurus could really operate a computer at that time – or so people thought.

Computer and software visionaries probably understood Lisa's operation and utility, but they were a tiny minority of potential users. A few years later, the Mcintosh *did* capitalize on the mouse and graphical file management.

This case study demonstrates that a cost-effective solution relies on understanding, as well as pricing. This failure of the Lisa falls into the Business/Political arena. It tends to be very difficult to predict and avoid these kinds of failures.

Case Study 6: Terrier LEAP and Aegis LEAP

The U.S. Navy ran the Terrier LEAP (light exo-atmospheric projectile) experiment between 1991 and 1995 to demonstrate the feasibility of intercepting ballistic missiles. The concept was to show that early 1990s technology could detect, track, and destroy a tactical ballistic missile.

The target for Terrier LEAP was a modified second stage from a Minuteman intercontinental ballistic missile. It emulated a SCUD, the Soviet derivative of the old German V2. The interceptor was a modified U.S. Navy Standard Missile with a third-stage booster and a kinetic kill vehicle. A kinetic kill vehicle destroys a warhead in flight by smashing into it, much like a bullet hitting a bullet. Missile boosters loft the kinetic kill vehicle into intercept range. The kinetic kill vehicle then coasts at very high speed to the intercept using a complex sensor suite and processing to detect the target. The intercepts were to occur outside the earth's atmosphere over the open ocean.

The Terrier LEAP experiments launched targets from Wallops Island, Virginia, and launched interceptors from a guided missile cruiser in the Atlantic Ocean. Large radars on Wallops Island tracked the targets and a system developed by The Johns Hopkins University Applied Physics Laboratory transmitted the missile position to the cruiser. [19]

The high-resolution radars at Wallops Island had to track the targets because the shipboard radars had insufficient resolution to pinpoint them. The processor at Wallops Island collected data from the radars, filtered the target track with a six-state Kalman filter, and transmitted the tracks to the ship. The target tracks were sent to the ship over telephone landlines and Inmarsat satellite links. The processor on the ship received the data, predicted the intercept time and point, and triggered the ship to launch its interceptor missile.

Testing finds some problems

We found a number of problems through extensive testing of the system. We performed end-to-end tests of the system by simulating a target launch, transmitting the simulated data through the entire system to the ship, and then calculating an intercept as if we were at sea. These tests found a number of problems, which, after correction, allowed my team to develop an effective system for target tracking.

Testing proved that we had to physically separate the landlines, so that one ran through New Jersey, the other one through Pennsylvania. Originally, the two redundant landlines ran together through New Jersey but the telephone company interrupted an early test when it brought down an intermediate switch in the telephone network for maintenance. Both lines were dropped in the interruption; this incident forced us to use separate landlines. We used separate satellites in the Inmarsat system for similar reasons.

Two shipboard radars caused problems for the LEAP system. The SPS-49 radar jammed the Inmarsat receivers while shipboard radar, the SPS-20, jammed the GPS receivers. We found both these problems only through testing the system. Consequently we had to ensure that both radars were off during the actual intercept attempts.

The signal algorithms within the modems greatly affected the robustness of data transmission. Originally, we used standard commercial modems but we found that the characteristics of the satellite links caused numerous pauses, retrains, and dropouts in them. We switched to modems that were designed for cellular telephones because their proprietary algorithms handled fading and hand-off between cells very well.

They performed much better than the original modems and greatly reduced the rate of data dropout in the satellite communications environment.

Two problems occurred in the signal processing. The Kalman filter had constraints that were initially too tight. It lost the target track when wind gusts jostled the radar's antenna thereby increasing the noise in the radar data. A second problem occurred in the servo algorithm that controlled the radar's antenna pointing. It had incorrect gains and it responded with overshoot and oscillations in the pointing. Again, we discovered both these problems through extensive end-to-end testing of the targeting system.

Another problem was the simple matter of time-tagging the data from the radars. The radar data had global positioning system (GPS) format but the system used universal time code (UTC) format. When we tried passing GPS time into the Navy's weapon system that expected UTC, we found that it prevented the calculation of valid solutions for intercept because the target times did not match the weapon system's times. Furthermore, when the UTC rolled over for either midnight or end-of-year change, it caused problems with time tagging of the data.

Results

Two intercepts were attempted; both missed. The first attempt failed because of an error in programming the booster of the interceptor. The second attempt failed when an improperly installed pyrotechnic switch, called a squib, failed to provide electrical power to the kinetic kill vehicle. Simple human error caused both of these failures. Somehow the appropriate tests or inspections did not take place to find these problems before the actual experiment.

What we can learn from Terrier LEAP

While many components of the LEAP experiment worked flawlessly, the primary objective failed. Once again, careful, thorough, and comprehensive development tests did not occur for all components or for the final, entire system. [20] Simple human error caused both failures. Ultimately, all failures come down to human error or inadequacy.

Background for Aegis LEAP [21]

Beginning in the mid-1990s, the U.S. Navy began system design of a ballistic missile defense, called Navy Theater Wide. It followed on Terrier LEAP and its purpose was to defeat medium- and long-range missiles. It uses redesigned components of the current air defense system: the AN/SPY-1 radar to acquire and track the missiles, the ship's weapon control, and the SM-3 (variant of the Navy Standard Missile – hence the label SM) interceptor missile, which is a four-stage derivative of the SM-2 Block IV missile.

The system is in the midst of a series of nine flight tests with intercepts of target missiles. The targets launch from the Hawaiian island of Kauai while the ship, the USS Lake Erie CG 70, launches the interceptors from 300 miles out (about 500 km). These are difficult tests because the interceptor is flying two times faster and five times higher than the current air defense missile. The kinetic warhead (KW) also must fuse various data, such as the ship's radar track of the target, GPS position, and the missile's inertial reference.

A robust ground test program provided end-to-end tests and a "methodical piece-wise evaluation using a variety of tests and test devices." [22] These tests included the following:

- Separation tests that included the squibs (pyrotechnic switches), batteries, and explosive bolts,
- KW hover test for the closed loop control of pointing towards the target,
- Air bearing tests to demonstrate various maneuvers: pitch-to-ditch the nosecone, IR seeker calibration against deep space, and pointing at the target before separation,
- Hardware-in-the-loop simulation and test of the avionics suite,

- KW tests for the IR seeker characterization, line-of-sight stabilization, third stage interfaces,
- Altitude testing in a vacuum chamber for circuit board delamination, corona and arcing, and volatile materials outgassing,
- Aerothermal testing in a hypersonic wind tunnel for nosecone heating and outgassing, seeker shield function, and strake heating and insulation.

These ground tests proved useful; many concerns were discovered, addressed, and solved. Following the ground tests, three flight intercepts successfully occurred in 2002. [23 – 25] The program continues with five more intercepts planned.

What we can learn from Aegis LEAP

The thorough and careful process of architecting, design, and testing has made Aegis LEAP a successful program thus far. It stands a far better chance of becoming a useful and viable missile defense system than systems that preceded it.

Recalls – products and automobiles

Product recalls are a form of failure. They cost companies money and reputation, and sometimes threaten their very existence. We can learn from product recalls. Here are some examples: [26]

- [Company A] recalled 45,000 heaters for defective thermostats that were improperly positioned, which could lead to the overheating.
- [Company B] recalled 3.1 million dishwashers. The slide switch (the lever that selects between heat drying and energy saving) can melt and ignite over time, posing a fire hazard.
- [Company C] recalled 5,500 toy flashlights because the batteries may overheat or leak and children can suffer burns from the leaking battery.
- [Company D] recalled upright vacuum cleaners because the power cord may break inside of the handle posing electrical shock and burn injury hazards.

Automobiles are complex systems and products. Their recalls can be instructive too. Here are some examples: [26 – 28]

- *March 12, 2002* [Company E] recalled the [specific model] trailer hitch – circuitry in the converter is inadequate to properly manage voltage spikes that can lead to an electrical short or open circuit within the converter, causing a failure and an inoperative trailer light.
- *September 11, 2000* [Company F] recalled about 270,000 [cars] – air bags that may deploy unexpectedly because of corrosion in the inflator.
- *During 2000* [Company G] recalled ignition modules that could cause a car to stall. When the temperature of an ignition module rises above a certain value the chances of the module cutting out also increases.
- [Company H] recalled 263,000 1995-97 [vehicles] . . . The electronic control module for the airbag could corrode from water or road salt and then accidentally deploy the driver side airbag.
- [Company I] recalled 757,000 1992-97 [vehicles] because higher than specified electrical load through an accessory power feed circuit may cause a short circuit and allow current to flow through ground wiring. This could cause overheating and an electrical fire.
- [Company J] recalled 1995-97 [vehicles] because improperly routed wire harness for the air-conditioner may permit wires to rub together and short circuit, resulting in a blown fuse, dead battery, or fire.
- *December 11, 1998* [Company K] recalled 226 [electric vehicles] to reprogram the logic in the motor electronic control unit (ECU), which can mistakenly detect a failure of an electrical current

sensor at speeds above 50 mph. It can cause the sudden loss of power and unexpected deceleration.

There are some common elements in these recalls.

- Passage of time – these were all fielded units.
- Nonobvious or obscure causes.
- Environmental interactions, i.e. corrosion, overheating.
- Failure modes with significant effects, i.e. fire or injury.

The nature of these problems share similar sources for each of the elements of failure. You could easily lump the nature and sources of these problems under human error. I choose to break them out separately:

1. The first characteristic of these problems is confounding complexity. The original designers did not foresee all possible circumstances and that many problems have multiple causes. In my first example, no one picked up on the possibility that wind could simultaneously down both power lines and telephone lines, which prevented protective mechanisms from functioning properly.
2. The second characteristic of these problems is oversight, i.e. unintentionally missing the potential problem during design, even if it appears to be a manufacturing problem. Improper routing of wire harnesses or manufacturing problems where wires are "stuffed" into an enclosure only to get crimped by hasty assembly or to chafe against a sharp edge is a design problem.
3. The third characteristic of these problems is nonobviousness to the user. The use of the device is improperly labeled or poorly instructed by the users manual or incomprehensible – knobs are in the wrong place or obscure codes must be programmed or it is awkward to handle.
4. The fourth characteristic of these problems is improper use. Yanking on a power cord repeatedly to unplug it from the wall receptacle, pounding on the keyboard out of frustration, poking the touch screen with a screwdriver instead of a finger, using the wrong lubrication, and your three-year-old standing on the toaster as a stepstool are all examples of improper use and abuse.

Personal examples

So far, we have looked at big systems and their failures or at product recalls. Let's bring the consideration of failure closer to home and how we can individually deal with it. I will describe some examples from my own experience to bring this to a personal level for you. I am sure that you will be able to relate to some of my examples.

Personal example – technical failure

Years ago I helped design a portion of an ultraviolet camera for a satellite instrument. I was responsible for the automatic gain control for an image intensifier that formed the front-end of the ultraviolet camera.

Image intensifiers are plates of glass capillaries sandwiched between phosphorus screens. They operate on the photomultiplier principle to intensify the light collected by the optics. They have wide dynamic ranges, with gains up to a million or more, and they are highly nonlinear in their control.

Problem: My first design for the control of the image intensifier produced a video picture that repeatedly bloomed (bright light washed it out) and then collapsed to nearly black.

Background: My design used discrete logic with up-down counters to record the number of bright pixels within a video frame. Unfortunately, the control for the image intensifier was unstable for bright objects. The short development time (we actually flew wire-wrapped breadboards) rushed me and I did not fully simulate or analyze the control action.

Should'a: In retrospect, I "should'a" taken extra time to analyze and simulate the expected video scenes during design of the gain control system. I let the tyranny of the urgent rule me and I did not take time to consider all the aspects of the problem. The good news is that a year later I got the chance to redesign the automatic gain control of the image intensifier for another satellite. This time I did a much better job, the second design controlled the image intensifier appropriately.

Personal example – professional failure

Years ago, I was asked to finish programming a project while the original designer moved onto another project. I procrastinated and made a number of false starts on the work. Finally, I removed myself from the work and the original designer had to return and finish the job.

Problem: I did not complete my assignment.

Background: Management had asked me to take over the project. I quickly realized that this was a no-win situation for me. While I was asked to finish the work and there was a hint that I be a hero if I did, it seemed to me that I would not see any real recognition of my work. I saw it as setting precedence for helping others out of problems without getting to do significant work myself. I simply lost motivation.

Should'a: I "should'a" either not taken job in the first place or, if given no choice, plowed through assignment while finding another job. This way I could avoid the problem of setting precedence of helping out others without getting to do significant work myself.

Personal example – professional/business failure

I had a business deal where bigger issues, outside my control, set me up for failure. My personal performance was good; I performed with technical and professional excellence and maintained my integrity. Eventually, I was accused of bad stuff, which I had not done, and the deal fell through.

Problem: Business politics outside my control skewered the deal.

Background: The business deal was an interesting proposition and product. I was banking on some long-term relationships. Unfortunately, another party introduced an unknown quantity early in the deal and then manipulated situations. This was beyond my foresight and foreknowledge. Weirdness mushroomed.

Should'a: I "should'a" either not made deal in the first place or, left earlier before the weirdness got out of hand. This was a tough one because the potential outcome was very attractive but the downside was unknown until I got into the deal.

Note: Always deal with integrity or don't deal at all. This always holds true.

Personal example – business/political failure

I led the development for a satellite subsystem. NASA, our sponsor, took over the project and out of our control.

Problem: Government politics outside the control of my company pulled the project from us.

Background: The team spent six months performing trade studies to design the architecture of the subsystem. The work was done thoroughly and well. Meanwhile, several groups at NASA had fallen on lean times and wanted work for their engineers. NASA pulled the project back in-house for their engineers.

Should'a: There is no "should'a." We did a good job and the project was still pulled from us. These things happen.

Case Study 7: Sidewinder, a successful counterpoint

After looking at different kinds of failures, I now examine the development of the Sidewinder missile, which turned out quite successfully. It managed to sidestep failure so that it did not end up as a footnote in history.

The Sidewinder is a simple and highly reliable air-to-air missile. It is launched from one aircraft to destroy another aircraft. The Sidewinder uses an infrared detector to seek and home on heat sources such as jet engine exhaust. It is very cost effective, if not downright cheap. The story of the Sidewinder's development is instructive for circumventing failure while advancing the state of the art in technology. [29]

Design choices

William McLean began studying heat-homing guidance in 1947 while working for the U.S. Navy in China Lake, California. His central idea was a spinning mirror, which made it gyroscopically stabilized; it reflected infrared emissions onto a photocell. The photocell produced an error signal that drove coils to adjust the spinning mirror if the emissions moved off axis. This formed the basis for a servomechanism to guide a missile. McLean also decided that the missile must be extremely simple, sturdy, and inexpensive (only a few hundred US dollars).

In 1949, McLean gathered a small team and began prototyping efforts to test his concepts. Remarkable ideas germinated in the process.

A member of the team, Sidney Crocket, suggested "rollerons," solid wheels with notches in their rims and mounted on the flaps of each rear wing to prevent the missile from rolling. These rollerons and flaps opposed the rolling motion of the missile. They were deceptively simple and did not require any additional circuitry or mechanisms to stabilize an important aspect of missile control.

The team developed a design for proportional navigation, which means that the missile never worries about its current position. It just tracked the target and kept on a collision course. This type of navigation simplified the circuitry.

The team developed the torque-balance servo for maneuvering the missile, which automatically compensated for differences in speed and altitude when twisting the fins. The servo system generated a torque command, rather than a position command, to adjust the fins. At high speeds, or low altitude, a certain torque adjusted the fins by a small angle, whereas at low speeds, or high altitude, the same torque adjusted the fins by a larger angle to achieve the correct maneuver.

The missile incorporated canard fins on the front to turn it in flight. McLean designed it this way so that the missile could be easily disassembled for storage. This design avoided the need for wiring, and connectors, to control the fins near the rocket nozzle end. McLean understood that connectors were often the source of reliability problems.

Flight testing required radio telemetry for the data acquisition system. They used Heathkit radios, kits designed for amateur electronic hobbyists, to develop the telemetry system.

Development choices

In 1951, McLean's team got the go ahead to develop the missile. They did a number of things to make the development successful. They used a careful, step-by-step test regimen. They tested the airframe and

seeker separately and then together on the ground. Then they tried ground firings of the missile, followed by air firings. After that they performed air firings against drones and target rockets. All these were done without operational warheads. Finally, they had air firings with live warheads against drones.

The team developed instrumentation to find quick answers to important questions. They believed that test results, not theory, should decide the utility of each design choice.

Testing uncovered a number of problems that McLean's team then solved. The gyroscope wobbled when the infrared signal was too strong, so they designed a damper for it. Unbalanced rollerons generated a vibration that caused the hot gas power source to burn improperly. Balancing the rollerons fixed the problem. By September 1953, they had 12 unsuccessful shots of the missile. They prepared two missiles and instrumented both. First, they fired one, recorded the telemetry, and found that it lost the target signal when the rocket motor burned out. They then fixed the problem with stronger retaining pins and tested the second missile successfully on 11 September, 1953.

For test equipment, the team designed the Sidewinder so that it only needed a flashlight and a small multimeter. A competing Air Force missile, the Falcon, required a wall of dials, gauges, and meters 40 feet in length (12 m) for its tests.

Results

The Sidewinder had an average miss distance, against a point target, of about one inch (25 mm). It could knock small thermite flares off the wingtips of drones. Its accuracy often saved the expensive drones, which, after repair to their wingtips, could be reused.

In a fly-off test against the radar-guided Falcon missile, the Sidewinder performed superlatively well. The Sidewinders were mounted on the aircraft the night before the test and were left to sit overnight, something the finicky Falcon couldn't do. The first Sidewinder intercept went right up the exhaust nozzle of the drone. The second intercept was a downward trajectory over the hot sand that produced a glaring infrared signal for the background.

The Sidewinder was so simple and robust that eventually two seamen could retrieve it from storage and assemble it in about a minute. It spawned generations of new missiles that used its basic design principles: the ground-to-air Chaparral, air-to-ground Focus, ship-to-air Sea Chaparral, and the shoulder mounted Stinger missile.

What we can learn

Breakthroughs require vision and a visionary leader. A small team of people committed to working together and communicating facilitates effective development.

The Sidewinder also succeeded because McLean insisted on simple and robust design. It was well understood by all involved, from engineers to junior enlisted personnel who stored, assembled, and maintained the missile. Most importantly, the Sidewinder development team carefully prototyped and tested concepts before completing the formal design.

References

[1] H. Petroski, *To Engineer Is Human, The Role of Failure in Successful Design*. New York: Random House, Vintage Books, 1992, pp. 176 – 181.

[2] Ibid., p. 176.

[3] Ibid., p. 179.

[4] Ibid., p. 62.

[5] *Remarks during presentation by* M. Postek, "Nanotechnology Research Potentials at the NIST Advanced Measurement Laboratory," Greater Washington Nanotechnology Alliance, Special Topics Symposium, November 25, 2003.

[6] T. J. Kriewall, *Opening remarks at the Sixth Annual IEEE Symposium on Computer-Based Medical Systems,* Ann Arbor, Michigan, June 14, 1993.

[7] J. Ganssle, Managing Embedded Projects, *Embedded Systems Conference,* November 1998.

[8] "Ariane 5 Loss Avoidable With Complete Testing," *Aviation Week & Space Technology*, pp. 55 – 57, Sept. 16, 1996.

[9] "Ariane 5 Report Details Software Design Errors," *Aviation Week & Space Technology*, pp. 79 – 81, Sept. 9, 1996.

[10] P. Sparaco, "Board Faults Ariane 5 Software," *Aviation Week & Space Technology*, pp. 33-34, July 29, 1996.

[11] "Ariane 5 Loss Avoidable With Complete Testing," *Aviation Week & Space Technology*, p. 57, Sept. 16, 1996.

[12] N. G. Leveson and C. S. Turner, "An Investigation of the Therac-25 Accidents," *IEEE Computer,* pp. 18-41, July 1993.

[13] ibid., p. 29.

[14] D. Dorner, *The Logic of Failure, Why Things Go Wrong and What We can Do to Make Them Right.* New York: Metropolitan Books, Henry Holt and Company, 1996, pp. 29 – 33.

[15] ibid., p. 31.

[16] ibid., p. 30.

[17] ibid., p. 33.

[18] ibid., pp. 33-34.

[19] K. R. Fowler, "Instrumentation for Ballistic Missile Defense: Lessons Learned from the LEAP Experiment," *IEEE Transactions on Instrumentation and Measurement,* vol. 47, no. 5, pp. 1092 – 1095, Oct. 1998.

[20] E. Rechtin, "The Synthesis of Complex Systems," *IEEE Spectrum,* vol. 34, pp. *51—55,* July 1997.

[21] G. A. Sullins, "Exo-atmospheric Intercepts: Bringing New Challenges to Standard Missile," *Johns Hopkins APL Technical Digest,* July-September 2001, Vol. 22, No. 3, pp. 260 – 274.

[22] Ibid., p. 268.

[23] R. Wall, "Intercept Starts Long Road To Sea-Based Missile Defense," *Aviation Week & Space Technology,* February 4, 2002, pp. 34 – 36.

[24] R. Wall, "Missile Defense Test Complexity to Increase," *Aviation Week & Space Technology,* June 24, 2002, pp. 50 – 52.

[25] R. Wall, "Test Raises Bar for Future Intercepts," *Aviation Week & Space Technology,* December 2, 2002, pp. 30 – 32.

[26] http://www.matthewslawfirm.com

[27] http://www.crash-worthiness.com

[28] http://autorepair.about.com/library/recalls/

[29] R. Westrum and H. A. Wilcox, "Sidewinder," *American Inventions, A Chronicle of Achievements that Changed the World*. New York: Forbes, Inc., 1995, pp. 136 -143.