Human fallibility and automation: lessons of the Virgin Galactic crash

Sean Brady explores how the lessons of the 2014 Virgin Galactic crash are pertinent to all design engineers, irrespective of discipline.

Introduction

On the morning of 31 October 2014, Virgin Galactic's SpaceShipTwo detached from its transport vehicle, WhiteKnightTwo, and commenced a test flight in the earth's atmosphere[1]. SpaceShipTwo, a reusable suborbital rocket, was piloted by two very experienced individuals. Peter Siebold, the pilot, was 43 years old and had 49 hours' flying time in SpaceShipTwo; the co-pilot, 39-year-old Michael Alsbury, had 32 hours' flying time in the same vehicle.

Just after detachment from WhiteKnightTwo, SpaceShipTwo fired its rocket. It increased speed, approaching the sound barrier. Suddenly it became aerodynamically unstable. Then it broke apart. Its pilot, Siebold, along with his seat, was thrown from the vehicle. He managed to release himself from the seat, his parachute opened, and during the nine-mile drop to the ground he drifted in and out of consciousness. He suffered severe injuries. Tragically, Alsbury, the co-pilot, died in the crash and was found still restrained in his seat among the wreckage[1].

Virgin Galactic

By late 2014, SpaceShipTwo, together with WhiteKnightTwo, were the latest vehicles in Virgin Galactic's quest for space. WhiteKnightTwo is an aircraft shaped like a catamaran that climbs in a corkscrew manner to an altitude of about 50 000ft (~15 000m); during the climb SpaceShipTwo hangs beneath it (Figure 1).

At 50 000ft things really get interesting. SpaceShipTwo detaches from WhiteKnightTwo, with WhiteKnightTwo being borne upwards due to the loss of weight and quickly getting out of the way. In a typical flight, SpaceShipTwo fires its rocket and accelerates, turning almost vertically upwards. It accelerates past the transonic range (0.9–1.1 Mach) and becomes supersonic, pressing its two pilots and six passengers into their seats. Soon after the rocket shuts off and SpaceShipTwo's momentum carries it upwards following an arc, crossing the peak or apogee, before beginning its downward trajectory. As the vehicle traverses this arc it pitches over for passengers to have a view of the earth below, and they unbuckle for four minutes of weightlessness. Then, before gravity reasserts itself, they strap in, and the vehicle begins to gain speed as it descends.

The really clever part is how SpaceShipTwo re-enters the earth's atmosphere. Unlike the NASA space shuttle or Apollo Command Module, heat generation during re-entry is not a serious problem because SpaceShipTwo doesn't actually go into orbit. It does, however, have to slow down its decent (by generating drag) and it needs to ensure it remains facing the 'right way up' throughout. Generating drag, however, is a double-edged sword: while it's needed at re-entry, it needs to be minimised during the boost phase. SpaceShipTwo solves this conflict by using a feathered system – it changes its shape during different stages of the flight.

During the boost stage the feather remains un-deployed and drag is minimised, but during re-entry the pilot and co-pilot deploy the feather, which rotates through 60° and dramatically increases the drag on the vehicle (Figure 2). The deployed feather also automatically keeps the vehicle's underside facing downwards, thus simultaneously solving the orientation issue[1].

After re-entry comes the landing. As SpaceShipTwo is unpowered (with the exception of its rocket) it now behaves like a glider, with the pilots having one chance to land it safely – there is no power to abort and come back around for another pass. Controlling the vehicle in this phase is incredibly difficult, which is why only the very best aviators are accepted, many of whom are either ex-NASA or aviation test pilots.

Hiring the best people appears to

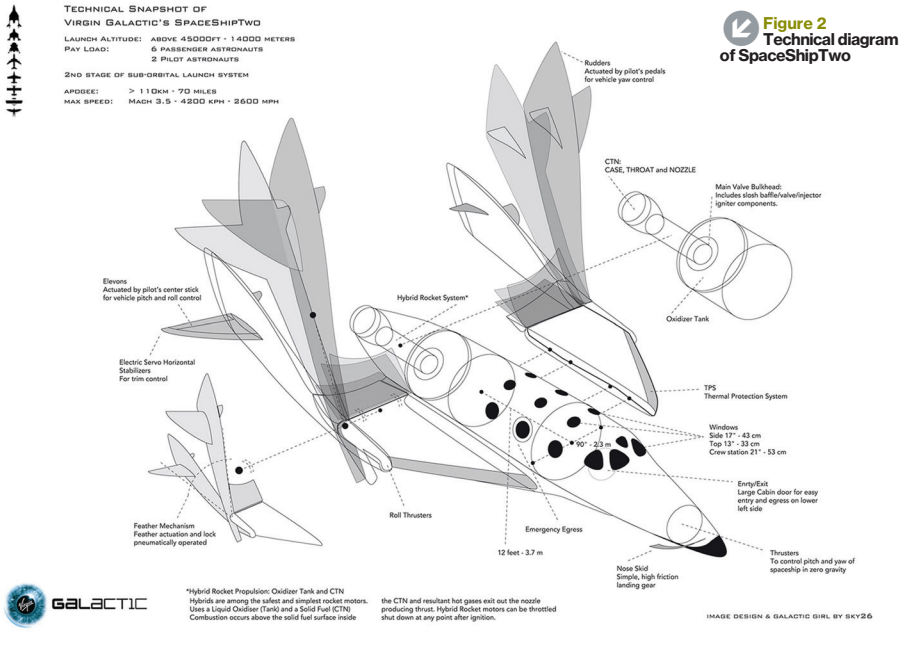

Figure 2 Technical diagram of SpaceShipTwo

VIRGIN GALACTIC

VIRGIN GALACTIC

Figure 1 SpaceShipTwo being carried by WhiteKnightTwo

have strongly influenced the vehicle's development, which was undertaken by Scaled Composites LLC – a subsidiary of Northrop Grumman, the same Grumman which built the iconic Apollo Lunar Module, the spidery vehicle that put Neil Armstrong and Buzz Aldrin on the moon. One of the philosophies adopted in SpaceShipTwo's design is that automation is minimised, with control being left with the pilots. (This is quite a departure from NASA's shuttle programme, where shuttle landings were wholly controlled by computer – in fact, it is actually doubted that a human could successfully land a shuttle unaided.[2]) Scaled Composites took the view that minimising automation also minimised the number of systems that could go wrong. Pilot intuition, reflexes and control would be the first and last line of defence.

The crash

On the morning of 31 October 2014, WhiteKnightTwo took SpaceShipTwo up to 46 400ft (14 142m)[1]. The test plan called for SpaceShipTwo to fire its rocket, then deploy its feather and glide back to the spaceport. It was the deployment of the feather that transformed the test into tragedy.

Feather deployment has two stages: unlocking and deployment. The co-pilot manually 'unlocks' the mechanical lock that keeps the feather in place, then both the pilot and co-pilot 'deploy' the feather by pulling two levers that activate actuators that rotate the feather through 60°. However, there is only a narrow window when the co-pilot can unlock the feather, which is when the vehicle is travelling between 1.4 Mach and 1.8 Mach.

Once above 1.4 Mach the aerodynamic forces acting on the feather prevent its deployment, and since the actuators used to achieve deployment are not designed to prevent deployment, the feather can be safely unlocked at this speed without fear of the feather deploying unintentionally. However, below speeds of 1.4 Mach, during the transonic range (0.9–1.1 Mach), the aerodynamic forces acting on the feather are such that they act not to prevent, but to cause, deployment. Therefore, unlocking below 1.4 Mach can result in accidental deployment.

The maximum of 1.8 Mach exists for safety reasons, providing a safe abort speed should the locking mechanism malfunction and the feather remain locked. If the feather is not unlocked by 1.8 Mach, then the pilots have to abort the mission. Aborting at this speed, by shutting down the rocket and minimising the height or apogee the vehicle would attain, means they can mitigate the hazards of re-entering with an un-deployed feather. Thus, if the pilots attempt to unlock before 1.8 Mach, and a malfunction presents itself, there is time to abort the mission and re-enter safely.

On the morning of 31 October 2014, SpaceShipTwo reached 0.8 Mach, and the forward-facing cockpit camera and flight data indicate that the co-pilot, Alsbury, called out the airspeed as "0.8 Mach"[1]. He then moved the feather from the 'locked' to 'unlocked' position. Thus, unlocking occurred not at 1.4 Mach, but at about 0.82 Mach, in the transonic range, when the aerodynamic forces act to deploy the feather. These forces were sufficient to overcome the capacity of the deployment actuators, which occurred quickly after unlocking[3]. The increased drag on the vehicle during this phase of flight resulted in it losing aerodynamic stability and breaking apart, as it was essentially folded in half.

Human factors

The National Transportation Safety Board (NTSB) investigation found that the co-pilot had spent many hours in the simulator, where he had repeatedly unlocked the feather at the correct speed of 1.4 Mach. So why did he unlock it at 0.82 Mach during the test? While we are unlikely to ever know precisely why, the NTSB identifies a number of issues that likely affected his performance on 31 October.

Firstly, from a physical perspective, flying SpaceShipTwo was quite different to being in the simulator. During the actual flight the pilot and co-pilot were subjected to significant G-forces and vibration that were absent in simulations.

Secondly, there was the workload, which was intense. Over a short period of time the pilots were required to perform a significant number of tasks from memory, with the NTSB concluding that such a high-pressure environment was likely to produce human error – even if tasks had previously been performed successfully in a simulator.

Thirdly, the fear of having to abort the mission if the feather wasn't unlocked by 1.8 Mach might have resulted in pressure on the co-pilot to unlock early. But despite this pressure, was the co-pilot not aware that there was a risk of catastrophic failure from early unlocking? It transpires that Scaled Composites was very aware of the catastrophic consequences of early deployment during the boost phase, but the NTSB found that "there was insufficient evidence to determine whether the pilots fully understood the potential consequences of unlocking the feather early"[1].

Which raises the most perplexing question of all: given the known catastrophic outcome, why did Scaled Composites not provide some form of automated system to prevent early unlocking? Disturbingly, the NTSB would find that Scaled Composites LLC did not include such a system because **it simply never envisaged that such qualified pilots would make such a mistake.**

The NTSB investigation would conclude

that "the probable cause of this accident was Scaled Composites' failure to consider and protect against the possibility that a single human error could result in a catastrophic hazard to the SpaceShipTwo vehicle.
This failure set the stage for the copilot's premature unlocking of the feather system as a result of time pressure and vibration and loads that he had not recently experienced, which led to uncommanded feather extension and the subsequent aerodynamic overload and in-flight breakup of the vehicle"[1].

Lessons

The key lesson from this failure is not new. It is succinctly summarised by James Reason: "It is often the best people who make the worst mistakes"[4]. The philosophy of minimal automation in the design of the vehicle left a critical vulnerability: no capability to prevent and manage a human error. It was foremost a system failure – it ignored human fallibility, a constant threat regardless of the expertise and experience of the individuals involved.

Ironically, the cause of this crash was diametrically different to the cause of the Hartford Civic Center Stadium roof collapse discussed in August[5]. In that case, rather than a lack of automation being the issue, it was an overreliance on automation (in the form of computer software) that caused the failure. The key issue here is how to ensure the reliability of human judgement: a lack of automation permits the inevitable human errors to occur; but over-automation encourages overconfidence in a system's infallibility and relegates human intuition to the side lines. I'm reminded of a (slightly modified) comment by David Brosnan when he wrote about the 'human' role in both failure and failure prevention: Human error may be the most important cause of failure, but human judgement may be our best safeguard against it[6]. Between these two extremes is a line we must all tread.

Sean Brady is the managing director of Brady Heywood. The firm provides forensic and investigative structural engineering services and specialises in determining the cause of engineering failure and non-performance.
Web: www.bradyheywood.com.au
Twitter: @BradyHeywood

REFERENCES:

1) National Transportation Safety Board (2015) *Aerospace Accident Report NTSB/AAR-15/02: In-Flight Breakup During Test Flight, Scaled Composites SpaceShipTwo, N339SS, Near Koehn Dry Lake, California, October 31, 2014* [Online] Available at: www.ntsb.gov/investigations/AccidentReports/Reports/AAR1502.pdf (Accessed: August 2015)

2) Hall J. L. (2003) 'Columbia and Challenger: organizational failure at NASA', *Space Policy*, 19 (4), pp. 239–247

3) National Transportation Safety Board (2015) *Video shown during NTSB Board Meeting on in-flight breakup of SpaceShipTwo near Mojave, CA* [Online] Available at: www.youtube.com/watch?t=15&v=Qv8Y0aMNix8 (Accessed: August 2015)

4) Reason J. T. (1990), *Human error*, Cambridge, UK: Cambridge University Press

5) Brady S. (2015) 'Hartford stadium collapse: why software should never be more than a tool to be used wisely', *The Structural Engineer,* 93 (8), pp. 20–22

6) Brosnan D. P. (2008) 'Human Error and Structural Engineering', *Structure*, September, pp. 46–49