At 9:34 am on Tuesday, June 4, 1996 an Ariane 5 rocket was launched on its maiden flight from the Guiana Space Centre. It veered off course 37 seconds after launch, began to disintegrate from high aerodynamic load, and was destroyed by its automatic flight termination system. The rocket and its payload, four European Space Agency satellites, had a combined value of $360 million, which was not insured.

An article by James Gleick in the New York Times some five months later provides a very understandable account of what happened:

“It took the European Space Agency 10 years and $7 billion to produce Ariane 5, a giant rocket capable of hurling a pair of three-ton satellites into orbit with each launch and intended to give Europe overwhelming supremacy in the commercial space business.

“All it took to explode that rocket less than a minute into its maiden voyage last June, scattering fiery rubble across the mangrove swamps of French Guiana, was a small computer program trying to stuff a 64-bit number into a 16-bit space.

“One bug, one crash. Of all the careless lines of code recorded in the annals of computer science, this one may stand as the most devastatingly efficient. From interviews with rocketry experts and an analysis prepared for the space agency, a clear path from an arithmetic error to total destruction emerges.

“To play the tape backward:

“At 39 seconds after launch, as the rocket reached an altitude of two and a half miles, a self-destruct mechanism finished off Ariane 5, along with its payload of four expensive and uninsured scientific satellites. Self-destruction was triggered automatically because aerodynamic forces were ripping the boosters from the rocket.

“This disintegration had begun an instant before, when the spacecraft swerved off course under the pressure of the three powerful nozzles in its boosters and main engine. The rocket was making an abrupt course correction that was not needed, compensating for a wrong turn that had not taken place.

“Steering was controlled by the on-board computer, which mistakenly thought the rocket needed a course change because of numbers coming from the inertial guidance system. That device uses gyroscopes and accelerometers to track motion. The numbers looked like flight data -- bizarre and impossible flight data -- but were actually a diagnostic error message. The guidance system had in fact shut down.

“This shutdown occurred 36.7 seconds after launch, when the guidance system's own computer tried to convert one piece of data -- the sideways velocity of the rocket -- from a 64-bit format to a 16-bit format. The number was too big, and an overflow error resulted. When the guidance system shut down, it passed control to an identical, redundant unit, which was there to provide backup in case of just such a failure. But the second unit had failed in the identical manner a few milliseconds before. And why not? It was running the same software.

“This bug belongs to a species that has existed since the first computer programmers realized they could store numbers as sequences of bits, atoms of data, ones and zeroes: 1001010001101001. . . . A bug like this might crash a spreadsheet or word processor on a bad day. Ordinarily, though, when a program converts data from one form to another, the conversions are protected by extra lines of code that watch for errors and recover gracefully. Indeed, many of the data conversions in the guidance system's programming included such protection.

“But in this case, the programmers had decided that this particular velocity figure would never be large enough to cause trouble. After all, it never had been before. Unluckily, Ariane 5 was a faster rocket than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, actually served no purpose once the rocket was in the air. Its only function was to align the system before launch. So it should have been turned off. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.”

The official report on the incident provides a few more technical details.

- The design of the inertial guidance systems (SRI) on the Ariane 5 was nearly identical to those used on the earlier Ariane 4. The software for the SRIs (indeed, for the whole flight control system) was written in Ada. Both SRIs operate simultaneously as primary and hot standby. If the primary fails, the control computer switches to the standby. Much of the software for the Ariane 5 SRI was reused from the Ariane 4.

- Ariane 5 is more than five times faster than Ariane 4 and uses a different launch trajectory. **Information about that new trajectory was not included in the requirements specification for the SRI given to the manufacturer**.

- No testing of the SRIs were done as elements of the whole guidance system, since they were considered to be prequalified by previous flights of Ariane 4. Tests after the failure using Ariane 5 trajectory data reproduced the problem exactly.

- After analysis, three of seven floating point to integer conversions were deliberately not protected by exception handling code since there was a desire to keep utilization of the control computer below 80%. The conversions not protected were believed to have values limited by physics to preclude overflows. This belief was not valid for the variable that did overflow because of the significant differences of trajectory between Ariane 4 and Ariane 5. **Software developers were likely not aware of the difference since the changed trajectory was not part of the requirements specification.**