The Chinook Helicopter Disaster

Prof. Simon Rogerson

Originally published as ETHIcol in the IMIS Journal Volume 12 No 2 (April 2002)

On Friday 2 June 1994 a Chinook helicopter ZD576 crashed on the Mull of Kintyre killing 29 people. Pilot error was suggested as the cause. Computer Weekly has been campaigning for this decision to be overturned as there was evidence that the crash could be a result of systems failures on the helicopter. On Wednesday 6 February 2002 the House of Lords committee report found that there is doubt about the cause of the crash because of the possibility of a technical malfunction. Extracts from various press articles and reports illustrate the broader issues related to the development of safety critical software development.

> In 1994, 29 people, including some of the UK's top anti-terrorism experts, were killed when Chinook helicopter ZD576 crashed on the Mull of Kintyre. In the absence of any clear explanation for the accident, the crash was blamed on the pilots of the helicopter who were killed in the crash.
>
> Since 1997 Computer Weekly has published details of the compelling evidence that has emerged that the crash could, in part or in full, be a result of systems failures on board the helicopter.
>
> In 1999 Computer Weekly also published RAF Justice which made clear how the Royal Air Force had carried out a cover-up and blamed the pilots unfairly.
>
> Yesterday the House of Lords endorsed Computer Weekly's findings, putting pressure on the government to overturn the Ministry of Defence's verdict of gross negligence against the pilots of Chinook ZD576. While it is imperative that justice is done, it is also vital that the Chinook saga is recognised as an example of just how the severe the consequences of poor IT project management can be.
>
> —Victory! Lords confirm CW stand, Thursday 7 February 2002

> The Lords committee report, which names Computer Weekly as having provided information to Parliament, found that there is doubt about the cause of the crash because of the possibility of a technical malfunction, such as a jam of the pilot's controls or a sudden engine surge, caused by the Chinook's safety-critical full authority digital engine control (Fadec) system.
>
> **The Lords committee's verdict.** "We have considered the justification for the air marshals' finding of negligence against the pilots of ZD576 against the applicable standard of proof, which required 'absolutely no doubt whatsoever'. In the light of all evidence before us and

having regard to that standard, we unanimously conclude that the reviewing officers were not justified in finding that negligence on the part of the pilots caused the aircraft to crash."

Software problems. "It is clear that at the time of the crash there were still unresolved problems in relation to the Fadec system of Chinook MK2s."

Boeing's simulation. "We consider that Boeing's conclusions cannot be relied upon as accurate." (Boeing's simulation was crucial to the 1995 enquiry's conclusion that the pilots were in control.)

—Software flaw could have caused Chinook crash, Tony Collins, Thursday 7 February 2002

Be on top of the project, or you will be almost entirely in the hands of the supplier if there is a disaster. When a user company passes day-to-day control of a project to its IT supplier and then suffers a serious software-related problem or even a major software-related fatal accident, there may be no sure-fire way of establishing what has gone wrong or why.

This is because manufacturers cannot be expected to indict themselves after a major incident by identifying defects in their product or management of a project.

The problem with software is that only the manufacturer may understand it well enough to know what has caused or contributed to a crash; and even if the supplier wants to tell the whole truth, will its lawyers let it?

One solution may be to employ independent experts to scrutinise the manufacturer throughout a project - it may be too late to employ them after a disaster.

—Lessons to be learned from Chinook tragedy, Tony Collins, Thursday 7 February 2002

The most worrying single facet of the crash is that the Ministry of Defence and the RAF assumed that a lack of evidence of malfunction points to operator error being the cause.

What if a software problem caused the accidental firing of a missile that destroyed a town? What if a design error in a software-controlled train set off a complex sequence of events that caused a fatal crash? What if dozens of people were killed in a fire in a tunnel because software-controlled sprinklers failed?

The chances are that if software caused any of these accidents, we would never know. This is because when software fails, or it contains coding or design flaws, and these defects cause a major accident, there will be no signs of any software-related deficiency in the wreckage.

And only the manufacturer will understand its system well enough to identify any flaws in its design, coding or testing. Yet no commercial manufacturer can be expected to implicate itself in a major software-related disaster. So, if software kills large numbers of people it is highly likely that the cause of the accident will never be known.

This is especially likely to be the case if the software has failed in no obvious way, such as when a coding error has set off a chain of complex events that cannot be replicated after a disaster.

But after a major accident, convention dictates that someone must be blamed. Step forward the vulnerable equipment operators: the pilots, keyboard clerks, train drivers or anyone who cannot prove their innocence.

This is particularly so because the manufacturer, in proving its equipment was not at fault, may have millions of pounds at its disposal. It will also have the goodwill of the customer, which bought the highly specialised equipment and relies on the manufacturer's support for its maintenance.

In contrast, individuals - the system operators - may have minimal resources: no access to the manufacturer's commercially sensitive information, none of the manufacturer's knowledge of how the systems work, and little money for expert reports and advice.

Therefore, the weakest link after a major fatal accident will always be the operators - particularly if they are dead.

That is why the loss of Chinook ZD576 is so much more than a helicopter crash. To accept the verdict against the pilots is to accept that it is reasonable to blame the operators if the cause of a disaster is not known.

If we accept this dangerous principle we may as well say to manufacturers of safety-critical software, "We recognise you will try to do a good job, but if you create poorly designed, haphazardly developed and inadequately tested safety-critical computer systems that kill people, we acknowledge that you will never be held to account."

—Clear and present danger: why CW refused to give up on Chinook, Karl Schneider, Editor, Computer Weekly, Thursday 7 February 2002

A Boeing simulation of the last moments of flight took no account of the helicopter's new Full Authority Digital Engine Control (Fadec) computer system.

In extreme cases, the Fadec could cause the Chinook's jet engines to surge suddenly, making it difficult to control the helicopter.

—Boeing simulation ignored Chinook's Fadec system, Tony Collins, Thursday 15 November 2001

For reasons of safety, each FADEC was to have two "lanes" which performed similar functions. The main or primary lane was to be a computer system. The back-up, or as it called "reversionary" lane, was to be based on more conventional analogue technology.

But as time went on the project became more technologically ambitious and, without any opposition from the Ministry of Defence, the manufacturers went ahead with a system which was digital in primary and back-up mode. Unusually in a FADEC system, there was no mechanical backup.

— RAF Justice - a 140-page report on a cover-up of the Chinook's software problems published by Computer Weekly in 1999

A warning to IS professionals. The integrative nature of today's software means the lines are increasingly blurred between a product's components. This case illustrates the need to accept that those developing software have a joint obligation with other professionals in delivering safe, usable and trustworthy products. Project and post implementation management procedures must ensure products are effectively tested and monitored. Abnormal events must be thoroughly investigated and corrective action take. The fundamental principle must be to safeguard the public.

Please send your views on ethical and social responsibility issues and cases of ethical dilemmas to:

Professor Simon Rogerson
Director
Centre for Computing and Social Responsibility
Faculty of Computing Sciences and Engineering
De Montfort University
The Gateway
Leicester
LE1 9BH
Tel:(+44) 116 257 7475
Fax:(+44) 116 207 8159
Email:<srog@dmu.ac.uk>
Home Page: (http://www.ccsr.cse.dmu.ac.uk)