4. Fitness for Purpose

One problem we have is that software is most usually not declared to be fit for some particular purpose.

When did you last see a piece of software that was guaranteed to do some particular job?

More usually it's not guaranteed to do anything in particular.

It's more likely to be:
If you think this software is useful and you're prepared to pay your money, "Good luck to you!".

4.1. Windows 95 Limited Warranty

"You acknowledge that no promise, representation or warranty or undertaking has been made or given by Microsoft or any person or company on its behalf in relation to the profitability of or any other consequences or benefits to be obtained from the delivery or use of the SOFTWARE ....

You have relied upon your own skill and judgement in deciding to acquire the SOFTWARE ..."
How about that? It's your fault!

5. Public Brain Washing

Does your TV set have a reset button on the front?

Your computer does!

It's there because the software fails, not the hardware!

Furthermore, the general population has been educated to expect software to fail.

This is an extraordinary achievement.

But should we be proud of it?

We will return to some other aspects of this later.

6. Two Case Studies

We will look briefly at two case studies:

Ariane-5 rocket failure

Therac-25 "accidents"

7. Ariane-5 launch failure

CNN: Unmanned European rocket explodes on first flight
`http://www.cnn.com/WORLD/9606/04/rocket.explode/`

A Bug and a Crash, report by James Gleick.
`http://www.around.com/ariane.html`

Inquiry Board Traces Ariane 5 Failure to Overflow Error
`http://www.siam.org/siamnews/general/ariane.htm`

ARIANE 5 Flight 501 Failure, report by the Inquiry Board.
`http://java.sun.com/people/jag/Ariane5.html`

The Ariane 5 Flight 501 Failure—A Case Study in Systems Engineering for Computer Systems (INRIA)
`http://www.cse.unsw.edu.au/~se4921/PDF/the-ariane-flight-failure.pdf`

8. Therac 25

see IEEE Computer, Vol. 26, No. 7, July 1993, pp. 18-41

Computers are increasingly being introduced into safety-critical systems and, as a consequence, have been involved in accidents. Some of the most widely cited software-related accidents in safety-critical systems involved a computerized radiation therapy machine called the Therac-25. Between June 1985 and January 1987, six known accidents involved massive overdoses by the Therac-25 —with at least two resultant deaths and serious injuries. They have been described as the worst series of radiation accidents in the 35-year history of medical accelerators.

8.1. Genesis of the Therac-25

Medical linear accelerators (linacs) accelerate electrons to create high- energy beams that can destroy tumors with minimal impact on the surrounding healthy tissue. Relatively shallow tissue is treated with the accelerated

electrons; to reach deeper tissue, the electron beam is converted into X-ray photons.

The Therac-25 was a dual-mode linear accelerator that can deliver either photons at 25 MeV or electrons at various energy levels.

8.2. Accident history

Eleven Therac-25s were installed: five in the US and six in Canada. Six accidents involving massive overdoses to patients occurred between 1985 and 1987. The machine was recalled in 1987 for extensive design changes, including hardware safeguards against software errors.

Related problems were found in the Therac-20 software. These were not recognized until after the Therac-25 accidents because the Therac-20 included hardware safety interlocks and thus no injuries resulted.

8.3. The Operator Interface

The Therac-25 could shut down in two ways after it detected an error condition. One was a treatment suspend, which required a complete machine reset to restart. The other, not so serious, was a treatment pause, which required only a single-key command to restart the machine. If a treatment pause occurred, the operator could press the "P" key to "proceed" and resume treatment quickly and conveniently. The previous treatment parameters remained in effect, and no reset was required. This convenient and simple feature could be invoked a maximum of five times before the machine automatically suspended treatment and required the operator to perform a system reset.

Error messages provided to the operator were cryptic, and some merely consisted of the word "malfunction" followed by a number from 1 to 64 denoting an analog/digital channel number. According to an FDA memorandum written after one accident

The operator's manual supplied with the machine does not explain nor even address the malfunction codes. The Maintenance Manual lists the various malfunction numbers but gives no explanation. The materials provided give no indication that these malfunctions could place a patient at risk. The program does not advise the operator if a situation exists wherein the ion chambers used to monitor the patient are saturated, thus are beyond the measurement limits of the instrument. This software package does not appear to contain a safety system to prevent parameters being entered and intermixed that would result in excessive radiation being delivered to the patient under treatment.

Notice that we've also educated people to ignore error messages!!

Poor concurrent programming using shared variables.

Much poor programming.

No apparent awareness of the necessity to identify exceptions, and decide what to do with them.

9.1. References

Nancy Leveson and Clark Turner, An Investigation of the Therac-25 Accidents `http://cse.unsw.edu.au/~se4921/PDF/Other/Therac-25.pdf`

10. Challenge

The Therac-25 case is quite old, but, "What would you do to ensure that software you wrote would not cause the same problems?

This is a difficult question.

10.1. But Software Development is Difficult!

Yes, it is, but let's not make that an excuse.

How do you go about your programming assignments?

Do you imagine that if it fails you might be killed?

Perhaps you should!

Qualify assurance by (un)natural selection. A story.