THE MAGAZINE FOR THE IT PROFESSIONAL

iTNOW

SUMMER 2018

ETHICS

TECHNOLOGY / RIGHTS / RESPONSIBILITIES

bcs
The Chartered Institute for IT

bcs.org/itnow

INSIDE MEDICAL SOFTWARE: WHEN PROGRAMMING ERRORS COST LIVES

Harold Thimbleby, See Change Digital Health Fellow at Swansea University, takes a look at computer error in the health sector, and considers the challenges faced in ensuring a change for the better.

We make many errors because we don't notice them until it's too late. A momentary daydream or distraction can result in a tea bag being placed into a kettle instead of a cup, with the guilty party not noticing until after the fact. Fortunately, a tea bag in a kettle is not a hard error to recover from.

When we are programming, we make mistakes, and the consequences may not be visible for a long time. The users of the programs may not understand that their problems are triggered by faulty code. Nurses have to respond rapidly when under huge workload pressures; programmers can take years developing systems for hospitals, and they should use that time to anticipate and properly manage the task.

When a button is pressed on an electronic device, conductors move to make an electrical connection, which is recorded as a key press. The conductors usually bounce, perhaps 100 times in a millisecond before they settle down. Key bounce is a standard problem, and it must be solved for buttons to be reliable. A simple solution is to use electronics, but it is cheaper to connect the button directly to the computer and fix the key bounce in software. However, if the programmer programs it incorrectly, the program will have a bug. The programmer probably won't be aware they made an error, and the device will go into production.

Cardinal Health is a company that makes medical devices, where the code should have very few errors. Cardinal Health was issued warning letters by the FDA, the US medical device regulator, outlining key bounce problems with their pump. Then the FDA had to issue a Class 1 Recall (meaning there is a recognised risk of death), affecting 150,000 devices, and involving US Marshals seizing equipment worth $1.8 million.

One particular problem of note was when a patient received an over-infusion of oxytocin. The pump was intended to be set for 36 mL/hr but was set to a rate of 366 mL/hr, ten times higher. The single digit 6 bounced, and was recorded as two presses, making 66.

If, after a key bounce like this, there's an investigation, the pump's log will show the nurse entered (in this case) 366 mL/hr, making it look like they negligently entered the over-dose. In fact, the device malfunctioned.

If a nurse is charged with manslaughter after a fatal error, the key bounce bug can mislead the prosecutors, and the nurse may be persuaded into a plea bargain. Key bounce errors are hard to reproduce, and if the prosecution wants to check a device thoroughly they will probably send it to the original manufacturers — who have a conflict of interest!

If a nurse is charged with manslaughter after a fatal error, the key bounce bug can mislead the prosecutors, and the nurse may be persuaded into a plea bargain.

The price of errors

When we make errors we can be reluctant to admit them. It then seems highly unusual when an error does come to light. When Kimberly Hiatt, a critical care nurse, made a calculation error, she reported it, was escorted from her hospital, put on leave, investigated and fined. She was devastated and committed suicide.

When my father was killed by an error, the doctor's computer report said there would be full recovery. Yet dad was already dead. Had the doctor reported it honestly, he might have been treated like Kimberly Hiatt.

When Lisa Sparrow gave a patient 100mL with a drug infusion pump instead of 10mL, she was reported by the *Daily Mail* as a 'blundering nurse'. In her trial, it was claimed no error was found with the device

Image: Getty/NanoStockk

doi:10.1093/itnow/bwy054

she used, yet the hospital replaced all the pumps with 'safer ones'. The implication is that the original pumps were part of the problem.

Dr Hadiza Bawa-Garba was convicted of manslaughter when a child in her care died of sepsis. Her trial has been controversial because it sends out powerful messages around how error is blamed on good clinicians: She had an impeccable record. Yet almost unremarked is that there was an IT failure lasting four hours, which delayed her getting blood test results and probably caused other distracting chaos. Surely the programmer (or the cyberattacker?) is partly responsible for the manslaughter?

We can be confident Dr Bawa-Garba was trying to keep her patients alive despite hindrance from her IT, but if you read the 'warranty' and disclaimers on any software, you wonder whether programmers have anyone's interests at heart other than their own. Many EULAs (end user licence agreements) require the user to indemnify the manufacturer! That does not encourage them to write safe programs.

These are just a few examples, but what is the scale of the problem? Best estimates put preventable error as a top killer, comparable to cancer and cardiovascular disease. The rate of serious harm, rather than death, is estimated to be 20 times higher.

Every patient is managed and treated by computer, from booking appointments, handling tests, delivering drugs and more. Computers do have bugs, so computer-related harm — causing error, not stopping user error, not helping detect errors — must be significant.

If computers only contribute to 10 per cent, just that would exceed the annual deaths from car accidents. We worry about making roads and cars safer. We demand safety technologies: safety belts, air bags, ABS. So why don't we worry about making hospital computing safer? Contrast Cardinal Health's attitude to bugs with General Motors, who, in 2016, voluntarily recalled four million cars over a bug suspected of killing one person.

Often the nurse will agree to be scapegoated, because the computer evidence incriminates them. Who can argue with gigabytes?

The power of scapegoating has a lot to do with it. When an error happens, if the nurse or doctor is blamed, the problem seems solved. The hospital no longer has the 'bad nurse', and they have saved themselves costs of computer investigations, and they have saved themselves worrying that their expensive computers may be unreliable.

Often the nurse will agree to be scapegoated, because the computer evidence incriminates them. Who can argue with gigabytes? The spurious logic of scapegoating reinforces itself: if the nurse is to blame, then they have betrayed our trust, and if we are betrayed, we are justified blaming them. The blame culture reinforces itself by psychological mechanisms of displacement and denial.

Furthermore, the law is against the clinician: if the device has been CE marked, the presumption of error is caused by the user. And it is easy to get CE marks. There is no robust process.

Programming is difficult, and safety-critical programming is especially difficult. Yet medical programmers need no qualifications. To become an anaesthetist, if you pass the exams, it takes eight years. If you want to program a pump to deliver anaesthetics, you can start now with no exams. Anaesthetists have standard operating procedures. Programmers don't.

Blinded by science

People are excited by computers. The NHS wants to go paperless, and everybody wants to use blockchain to improve things. But there is no evidence it is effective.

Going from the lab to an approved drug can take 15 years. We understand how to develop drugs, do randomised controlled trials, and so on.

We have little idea how to develop programs and assess them for safety and effectiveness. If a drug takes 15 years to get to market, why are we rushing into new computer 'solutions' that have not been rigorously developed or tested? If somebody develops a new blockchain

technology for healthcare, shouldn't we develop it at least as carefully as a drug, and, if trials are successful, maybe start using it in 2033?

There is this assumption that the latest computers are an improvement, but speed and fancy technologies like blockchain (and cloud and big data and ...) is an addictive drug. If computers are perceived as perfect and something goes wrong (as it eventually will) then it logically follows that something else must have caused the problem. It must have been the user. If we scapegoat the user, the problem seems to be solved. Scapegoating is a deceptively simple explanation that saves us the daunting work of evaluating our IT. Disciplinary processes then satisfyingly make sure mistakes don't happen here!

Ways forward

I have only explained a few simple healthcare bugs. Many are much harder to spot; many, I think, are never spotted.

Modern healthcare is amasing and we entrust our lives to it, which makes it seem all the more shocking when anybody admits to problems or gets caught.

Scapegoating dedicated NHS staff is not going to help improve the system, though it gives a misleading impression of trying. We must be clear what has really gone wrong if we want to improve.

1. BCS, or equivalent technically authoritative organisations, should have a task group to evaluate any incident, so the right lessons are learned.
2. BCS should help the NHS procure safer systems and equipment. These ideas will put pressure on industry to improve, and — if they want to — there are many ways to improve, such as adopting software safety processes from aviation.
3. We should improve regulation to require appropriate evidence that healthcare software is dependable and that it actually delivers cost-effective benefits to patients.
4. We should licence and require safety critical systems programmers to be at least as competent as professionals working in the field.
5. When something goes wrong, every defence failed including computer systems — but blaming the programmer is as problematic as scapegoating the user.

These are some suggestions to start the conversation. We must start something before we have a thalidomide-scale computer-related incident that forces our hand, because when computers go wrong they can do it on a huge scale. A clinician can only kill one person at a time, but a programmer can kill thousands...

If we scapegoat the user, the problem seems to be solved. Scapegoating is a deceptively simple explanation that saves us the daunting work of evaluating our IT.

Acknowledgements

This article is based on a lecture sponsored by BCS's ICT Ethics Specialist Group, which took place on 6 March 2018.

Further reading

1. H. Thimbleby, A. Lewis & J. Williams, 'Making Healthcare Safer by Understanding, Designing and Buying Better IT', *Clinical Medicine*, 15(3):258–262, 2015. DOI 10.7861/clinmedicine.15-3-258 — a review of many healthcare IT problems and suggestions for better procurement processes.
2. H. Thimbleby & P. Cairns, 'Interactive numerals', *Royal Society Open Science*, 4(160903), 2017. DOI 10.1098/rsos.160903 — flaws and defects in number entry user interfaces are ubiquitous and easily avoided (once recognised).
3. H. Thimbleby, Cybersecurity Problems in a Typical Hospital (and probably all of them), *Proceedings of the 25th Safety-Critical Systems Symposium*, 415–439, 2017 — a review of a large criminal case based on flawed patient data.
4. H. Thimbleby, Trust Me, I'm A Computer, *Future Healthcare Journal*, 4(2):105–108, 2017. DOI 10.7861/futurehosp.4-2-105 — the psychology of IT misdirection.
5. M. Thomas & H. Thimbleby, Computer Bugs in Hospitals: A New Killer, *Gresham College Lecture*, 2018. www.gresham.ac.uk — transcript of a public lecture, providing an authoritative review of the problem. Includes extensive bibliography.