Safety versus Security in Healthcare IT

Harold Thimbleby

Swansea University

Swansea, UK

Abstract Safety and security are different but closely related concepts. Security is especially relevant in domains like finance, and safety is especially relevant in domains like healthcare and aviation. We review some of the safety problems besetting healthcare IT systems, and we show that some problems are technically open to improvement. Given that there are almost-free technologies (e.g., as part routine software upgrades) to improve safety, it is important to explore the reasons why healthcare safety does not get the priority that, for instance, security does in finance or safety does in aviation.

Cite as: H. Thimbleby, "Safety versus Security in Healthcare IT," in *Addressing Systems Safety Challenges*, Proceedings 22nd Safety-Critical Systems Symposium, C. Dale & T. Anderson, eds., pp133–146, 2014. ISBN 978–1491263648.

1 Introduction

Patient safety is a global priority (Donaldson and Philip 2004), and computers are surely part of the solution to improving healthcare. Yet healthcare has become a computing problem: computer systems under-perform and seem counter-productive. In contrast, in some domains like finance, computers have been enormously successful. Your and my bank accounts can be computerised the same way, but your health records are very different to mine. Nothing as simple as addition can summarise my patient records. Since finance is uniform but healthcare very dependent on the patient and needs of the practitioner (radiotherapist, consultant, pharmacist) some commentators have identified the much greater need for User Centred Design (UCD) in healthcare (Landauer 1996). But those comments were made in the last century, and international standards (such as ISO 14971 (ISO 2012) on the application of risk management to medical devices) are based on them. UCD (at least as interpreted by the ISO standards), or the lack of it, is not at the root of problems.

This paper reviews the context of healthcare safety as it relates to computing systems. We then contrast the state of healthcare with finance; the drivers are in-

terestingly different, and we suggest that the differences between safety and security are insightful.

2 Healthcare error

The World Health Organization (WHO) defines 'never events' as errors that should not occur. An example is 'wrong site surgery' such as operating on the wrong patient or on the wrong part of the right patient. The idea is that never events do not and should not occur.

Yet approximately one in ten hospital patients suffer a preventable error. Obviously no professional makes errors deliberately, so errors must be unnoticed at the time they occur. A *preventable error* is defined as one that would have been avoided had all the known information been correctly interpreted and acted upon in the correct way. For example if you are allergic to a certain drug but nobody knew until your allergic reaction, giving you the drug would be an error only in hindsight, but it was not a simple preventable error. Often preventable errors are defined as being unprofessional, and are potentially criminal, for the nurses or clinicians involved.

Consider the following example. A nurse gives a patient 5 mg of morphine, and they are over-dosed and die. The consultant's prescription the nurse read said .5 mg, and by misreading the decimal point a dose ten times too high was administered. Perhaps the nurse thought the dose was high and asked a colleague to check it. Perhaps the second nurse also thought the prescription said 5 mg. Here the nurse has taken steps to prevent an error; the probability of an error occurring was reduced (though in fact it still occurred).

There are rules (ISMP 2007) for writing drug doses, and 'naked decimal points' (as in the decimal point in .5) are not allowed. So far as the nurse is concerned, they made an *induced error* – a recognised problem, misreading .5 as 5, occurred, and it should have been prevented by writing 0.5 mg or writing 500 mcg (1,000 mcg = 1 mg; note that μg is not used because it can be misread as mg). In other words, the environment the nurse worked in, in this case somebody writing .5, induced the error where 0.5 was misread as 5. So although we might not blame the nurse for the slip, we still blame somebody else, in this case the person who wrote down the prescription for not following standard procedures.

Imagine now that a computer is involved in the event. For example, the nurse read 5 mg off a computer screen rather than off a hand-written prescription. There is now an audit trail that shows the display really did indeed show 5 mg. There is also an audit trail to show the consultant keyed in 5 mg. The computer system is behaving as designed, so now it is clear that the consultant made an error. The consultant should have entered 0.5 mg but they entered 5 mg.

Let us imagine the consultant correctly planned to enter 0.5 mg. Unfortunately they hit • (the decimal point) and they accidentally clicked it twice. They noticed this keying error, and hence having keyed • twice, they pressed DELETE, then 5,

then hit SEND. Had they been using Microsoft Office, they would indeed have entered 0.5 at the end of this sequence, but they are using special-purpose software designed for patient records. The user interface has been badly designed and poorly implemented: pressing decimal points more than once is ignored, so pressing DELETE deletes the 'only' decimal point. Hence 0 • • DELETE 5 becomes 05 which is treated as 5. This is the source of the ten times error, but the logs of the consultant's activities show they simply entered 5 mg. The records 'prove' the consultant is to blame. An incautious consultant might incriminate themselves, 'I didn't think I entered 5 mg, but the logs say I did, so yes, I must admit I made the mistake.' The logs do not show what the system did, nor how it turned (in this case) a *correctly corrected* error into an actual error leading to patient harm.

We agree an error has occurred; moreover, the error led to a patient receiving a ten times overdose of morphine, and this (in the story) leads to death. A computer system is part of the story, but it has been certified for medical use *and it did not malfunction*. It did was it was designed to do. Therefore (it seems) the error *must* have been caused by the operators: here, the consultant. However, as we described the sequence of events, such reasoning would be flawed.

The culture in healthcare is that professionals are perfect, and they have been professionally trained. The computer systems are perfect too, and they have been certified. The computer systems worked as designed; therefore the nurse (or other professional) must have betrayed us. It is not unusual to find the media vitriolically reporting stories of 'witches' – angels who have gone bad.

Healthcare chooses to focus on error rather than patient harm. Some errors are unprofessional and are so-called never events. Yet many errors do not lead to harm, and even never events can be mitigated – for example, a surgeon making an incision in the wrong place need not escalate that error into removing the wrong kidney if the error is noticed and intercepted. Indeed, there is a movement to promote resilience, reducing the impact of error: in 'Safety I' the focus is on the errors, and often blaming the people making identifiable errors; in contrast in the more enlightened 'Safety II' the point is that 99.9% or more of the time there is no harm – so focus on making that 99.9% a larger proportion of the time. Instead of blaming bad things, support and encourage good things (Eurocontrol 2013).

As we showed above, if the focus is on blaming bad things, we have to be very sure we are blaming the right bad things – in our example, suspending or sacking the consultant misleadingly seems to solve the problem (the system has got rid of a 'culprit'), but in fact the system that induces the problem has remained unexamined and likely to induce further errors. Worse, the culture of identifying individual failures encourages users to keep quiet about any problems they encounter – which is a twofold problem. The organization cannot learn how to improve and take advantage of learning from these workarounds; secondly, if new computer systems are designed, they will be designed as if these workarounds do not exist (because nobody knows about them), and therefore they will perpetuate and perhaps exacerbate the error-inducing behaviours. More clinicians will be terminated on the high altar of software.

3 Healthcare versus finance

It is interesting to contrast the healthcare profession's approach to patient safety with the financial industry's approach to loss. The WHO says some events are never events; they do not and should not occur. There is no prioritising. In finance, one expects some loss, and the idea is to minimise it to acceptable levels – acknowledging that there may be vanishing returns in doing so. In particular, it is recognised that reducing loss is a cost, and there is a trade-off: ultimately there are vanishing returns from reducing loss.

Finance distinguishes between loss from external problems and loss from internal problems, such as fraud. There is no reason to ignore external loss (though one might not want to tell shareholders). In contrast, in healthcare 'fraud' is very rare, and there is no such thing as *external* loss – all preventable harm is presumed caused by staff. In other words, when patient harm occurs, somebody within the organisation is in principle responsible.

Patients in hospitals are ill – if they weren't they should have been discharged already. So it is unsurprising that patients catch infections and perhaps get worse and die through nobody's fault. It is then a short step to disguise some preventable errors as the inevitable consequence of a patient's illness: 'I'm sorry your father died. He was very ill and in his weak state caught an infection that killed him.' A longer story might have been: 'I'm sorry your father died. We forgot to change his central line, so he caught an infection from it, from which he died' (Pronovost 2010). So, most of the time the healthcare organization underestimates the rate of preventable error, because much harm is being disguised as inevitable.

When patient harm occurs that must have an explanation (e.g., because somebody has complained), it then appears that it is exceptional. Normally there are no 'errors' and so now there is an investigation uncovering errors, these must be exceptionally unprofessional. Perhaps an example then needs to be made out of the carer who made 'the mistake'.

4 Safety versus security

Safety and security are very similar concepts. Making a mountain climber safe means making them secure; conversely, making them secure makes them safe. A financier might talk about safe investments, making very little distinction between secure and safe. It is more helpful to make these words refer to clearly distinct concepts. Thus, *safety* is about achieving well-being, health and physical wholeness of people (or possibly animals or the natural environment more broadly); and *security* is virtual wholeness – only the right people have access to the information. Money rather blurs this distinction: because it is fungible it can be made either virtual or physical and still work just as well. Moreover, somebody who has very little money (e.g., who loses their money through a security breach) starts to

be at risk of starvation or other safety problems. It is not surprising the words have vague definitions retaining overlaps in various domains. Here, however, we are particularly interested in the use of these words for computer system design, and computers are increasingly making the distinctions hard: a door lock that provides physical security (with metal keys) can be implemented by a computer using virtual security (passwords and smart cards), and the lock may be used to ensure that people *securely* behind the locked door are protected from intruders and are therefore *safer*.

In a hospital, safety is primarily focussed on patient safety (staff safety is not much different in a hospital from any other workplace, apart from increased risk of illness). Security is about ensuring only authorised users have access to patient records; security in a hospital does relate to normal operations, and clinicians will therefore often take short cuts such as sharing passwords. In contrast, in a bank, the primary concern is security. The bank does not want to lose money to unauthorised people. Of course, staff safety is a major concern, but this is usually achieved through physical protections (such as door locks and thick glass screens) rather than by normal computer system operations.

5 Defining usability, security and safety for system designers

We can make some contrasts between the usability/security/safety concepts clearer by a narrower set of definitions based around the engineering decisions we have to make when designing computer systems. A philosophical or etymological debate is all very well, but how do we design more effective systems? As engineers, we want to improve the world.

- *Usability* is about making systems easy to use for everyone.
- *Security* is about making systems easy to use for designated people and very hard (if not impossible) for everyone else. It is about designing systems to be hard to use for the wrong people; we do not want the wrong people to do bad things with the systems. There are bad people, and we do not want them to easily access or use our systems.
- *Safety* is about stopping the right people doing bad things. Good people make slips and errors; in a safe system those errors should not escalate to untoward incidents, harm or other safety issues.

These simple and nicely contrasting definitions raise immediate tensions. A secure system may not be very usable. In turn this will encourage its users to find workarounds to make their daily use of the system more ‘user friendly’. Thus a hospital system may end up with everyone sharing the same password, as this is much easier than remembering individual passwords and logging in and out repeatedly. But making it usable for the right people has also made it usable for the wrong people (e.g., intruders), and thus undermines security.

The definitions seem to take it for granted what ‘good’ and ‘bad’ mean. But these are more subtle concepts. Good users do not intend their activities ever to be bad, but those actions may turn out to have bad consequences. So, safety is about designing for good people who may inadvertently set in motion things that end up going bad. Often users are dedicated to get their jobs done, and because the computer systems they use are not well designed, the users prioritise doing their jobs well regardless of obstacles the computers throw at them (Blythe et al, 2013). Their managers might think that their workarounds are ‘bad’ because they are not standard operating procedures, but often managers have no idea about what really goes on! So: although in the definitions the meanings of ‘bad’ and ‘good’ seem self-evident, in reality thinking that a good person does bad things is a short step away from scape-goating that well-intentioned person. What is really meant is that a good person does great things, but the system as a whole is broken and badly-designed, so a well-intentioned action leads irretrievably to a bad outcome. We gave an example of this above: a consultant corrects a typo, the wrong dose ends up being administered. ‘A good person did a bad thing’ clearly captures too little of the reality of the problem.

The number of users is an important consideration when designing systems. In the limiting case, the programmer is the only user, and they know how to use any program they develop. Hence programmers are very experienced at building user interfaces that are special cases. When systems are used by many users, usability problems often arise because the majority of their users are unlike programmers, and have no deep knowledge of the implementation. Furthermore, when systems are used by many users, some of them are more likely to be bad, so security becomes increasingly important.

The problem is that most programmers develop their experience and skills in the limiting case, and hence unconsciously under-play managing the issues that unavoidably arise in real systems. Good UCD practice requires experimental assessment of proposed designs, iterative development, with the actual users of the system.

A subtle consideration is the proportion of computers to users. In a typical financial application, the ratio is about 1:1. A user logs on to the secure system, and stays using it. In a hospital, the ratio is perhaps 1:20. Now users frequently have to log in and log out because the computer is shared. Since this is tedious and interferes with the task these users are performing (i.e., patient care), a common workaround is to log in one user, and then share their credentials for the remainder of the shift. Since passwords are often complex (so intruders cannot work them out) it is common for the passwords to be displayed prominently; while this helps the good users know the password, it has the side-effect of showing the password to potential bad users. Such workarounds make logging in easier and save logging in and out repeatedly, and improve the safety patient care. Ironically, the more extreme the computer:user ratio, the higher the overhead of logging in and out in proportion to the cost of doing tasks, and also, with more users, the higher the probability that there is a bad user wanting to gain access to the system. But security is not top of a carer’s agenda.

A series of people using medical systems without logging in and out can cause unfortunate problems. For example, most ECG monitors allow the sensitivity to be changed. If sensitivity is reduced by one operator, and the next operator is taking the ECG from a patient with left ventricular hypertrophy (LVH), a symptom of which is higher voltages, this problem will be very hard to notice. I don't know of any ECG monitors that bother to have logging in and out protocols and reset their settings in-between, because they are 'just' devices.

Whether workarounds in healthcare compromise security in a significant way is a matter for careful assessment. What is clear is that without proper UCD no designer would know – you have to watch the systems in use, not look at their logs. The fact that they appear to be being used successfully does not mean that the operating practices have not become risky, compromising both safety and security.

6 Expectations and culture

We have defined security as stopping bad people doing bad things. An engineer might use this definition to help think about passwords and so forth. But there is a more interesting angle: we *expect* bad people to do bad things. So when bad things happen, this is to be expected. We expect robbers to try to rob!

In contrast, safety is about stopping good people doing bad things. Again, an engineer might use this definition to think about confirmation dialog boxes, undo features, and so forth. But there is a more interesting angle: we do not expect good people to do bad things. It is more like a betrayal when they do. We do not expect nurses to kill patients.

Perhaps this is why financial systems are designed to protect against security breaches, but healthcare systems are not designed to protect against human error? Designers do not expect good people to be bad. Unfortunately, 'to err is human' – everybody makes errors eventually. This does not mean they are bad, but it does mean that the systems they use should be designed so that they are protected from their predictable errors.

However, the safety/security 'divide' cannot be the whole story: civil aviation safety is improving, but healthcare safety is not.

The US Institute of Medicine has made the memorable comparison that preventable deaths in US hospitals is as bad as a jumbo jet crashing and killing all passengers every week.

When an aircraft crashes the accident is visible, even, depressingly, photogenic. The consequences become publicly visible. From the airline's point of view, they have lost a very expensive piece of equipment – the airframe – and from their and the manufacturer's point of view potentially lost a lot of credibility and future market share. There is therefore high motivation to *correctly* seek out the causes of the incident so that there can be learning and similar incidents avoided in the future. For example, if the pilot was asleep, was that caused by inappropriate fly-

ing schedules? Certainly, blaming the pilot (so-called ‘pilot error') is a misdirection from the full story.

In contrast, in healthcare, the deaths (and other harms) occur one at a time and usually in private. Often the patient was seriously ill to start with, so their death may not be surprising – whereas death from an aircraft accident is surprising. It is easy to think of many simple reasons why a sick patient dies rather than seek the real causes (cf. the comments about central line infections above). The patient’s ill-health can often be blamed; whereas, except in cases of terrorism, it is not legitimate to blame the passengers for an aircraft accident.

When an aircraft crashes it is self-evident that the system failed to do what it is designed to do, namely to get an aircraft fly (and land) safely. When a patient dies, often the medical device involved (such as the linear accelerator that generated the radiation overdose) had been certified as compliant to relevant standards. It therefore seems logical that any error must have been due to the user if it was not the device. Some examples of this type of thinking are given in Thimbleby (2013) – unfortunately it often leads to inappropriate and indeed unproductive reprimands.

However, that a device is compliant does not mean it could not be better; and if it could be better, perhaps the incident in question might have been prevented by better design rather than being caused by operator error? We will give a concrete example of this possibility below: we show that a compliant system can be made safer; it follows that the relevant standards are not perfect. Some errors are induced by poor design.

So if there are many possible reasons why a patient died it is tempting to find the cheapest answers – namely fate or nurse error – and thus avoid seeking improvements in the system. In contrast, in aviation, the least costly option in the long run is to fix any failures in the system – blaming pilot error might be easy, but it won’t stop the next pilot making the same error. Moreover, if a second crash occurs for the same system reasons (e.g., a failure in the airframe design or in the airline’s operating procedures), when this becomes common knowledge the manufacturer or carrier will be even worse off. The profile of air accidents forces investigations to seek the truth; the personal nature of healthcare incidents encourages delay, deny and scapegoat.

7 Example – reducing the probability of out by ten errors

There may be cultural reasons for differences, but if some of the unsolved problems are not solvable then there are also engineering problems. If so, one would then have to seek solutions to improving healthcare in, for instance, better training or incentives. We now show that quite simple engineering interventions can make systems safer. One wonders why these ideas are not more widely used.

Let us assume the nurse notices an error with probability q; in our simple model we assume the error is either noticed immediately or not at all.

Example. The nurse should enter 5. With probability p they will press a random key other than 5 and if so, with probability q, they will press CLEAR or take other appropriate action (correctly using the particular user interface being modelled) to recover from that noticed error.

If the nurse notices an error, they will correct it, and it will not lead to harm. Unnoticed errors are problematic, since when they occur nothing in the system intercepts them. One might use training to increase q; one might design operating procedures that increase q – for example, having two nurses perform a calculation allows them to check each other, thus ideally increasing q to $1 – (1 – q)^2$ *except* that nurses are not independent, so the probabilities do not multiply up so nicely. For example, with a q of 0.5 each nurse we might hope the probability of noticing errors would improved to 0.75 by using a pair of nurses as 'buddies' to check each other. Unfortunately, if one nurse misses an error, the other nurse is likely to miss it too (many of the reasons the first nurse missed it will still apply to the second nurse, so the probabilities are not independent). Worse, human psychology means that if we think a pair of nurses is safer, we may take greater risks; or perhaps the second nurse does not actually check the first nurse's actions, but just feels the first nurse must have got it right. Some hospitals therefore require critical calculations to be done by a *single* nurse: since they have nobody else to rely on, hopefully this increases their attention to the details of the task and hence their q. This somewhat digressive discussion will motivate an alternative approach to improving q below.

Dose Error Reduction Systems (DERS) have been introduced for exactly this reason. A DERS knows the drug a patient is prescribed, and then limits the ranges of dosage to within safe values; effectively, it notices out-of-range errors and intercepts them. DERS have been shown to reduce patient harm.

The probability that the user makes a keying error is p, which is around 1% – that is, about 1 in 100 hundred keystrokes of an attentive nurse is in error.

With these assumptions, it is possible to simulate nurses entering data, making slips, and possibly intercepting those slips; similar methods have been discussed elsewhere (Thimbleby and Cairns 2010, Cauchi et al. 2013). One can then consider alternate designs and select those that let through fewer errors. One can also explore trade-offs; for example, it may be preferable to reduce significant errors preferentially, or if reducing unnoticed errors means significantly increasing the time it takes to enter data, in some cases (like resuscitation) where time is of the essence, some error latitude may be acceptable. The user interfaces analysed in this paper behave identically if the user makes no errors, and, in particular, the sequence of keystrokes required to enter a number correctly is identical in all cases.

A standard definition of significant errors for numerical data is an out by ten error. An out by ten error occurs when a number entered is ten times out (too high or too low) from the intended value. For almost all medication, an out by ten error results in significant patient harm (possibly including unnecessary pain and longer stays in hospital, etc.).

A Monte Carlo model assumes a user makes key slip errors with probability p, notices these errors with probability q, and corrects them (again potentially making further keystroke errors with probability p). We generate key sequences for numbers, and compare the intended and final values for several designs. Figure 1 shows typical results. The graph shows that the probability of out by ten errors decreases with increasing q for every design considered, but more interestingly it shows that designs differ significantly in how dependable they are. Furthermore, the safety ranking of the design assessments are independent of q. (In fact, but not shown here, the rankings are also independent of p or whether we measure 'out by 10' or 'out by k' for any $k > 1$.)

Three contrasting designs are chosen:

1. The user interface of the common HP EasyCalc 100 handheld calculator (introduced in 2009). This device has a DELETE key that is faulty: it deletes only digits and ignores decimal points (so that, e.g., 1 2 • DELETE is treated as 1).
2. A user interface with a correction key that works correctly.
3. A proposed user interface that enforces the Institute of Safer Medication Practices guidelines for writing drug doses (ISMP 2007), as discussed above.

The status of these three designs can be contrasted:

1. This is a general purpose calculator and as such may be found in hospitals for routine calculations, such as drug dosage.
2. Many user interfaces of this type have been certified for medical use.
3. No known system in use adheres to the ISMP guidelines.

For simplicity we do not consider here other design choices; for example, many user interfaces (like the HP EasyCalc 100) ignore multiple decimal point key presses. On these, pressing 0 • • 5 has the same effect as pressing 0 • 5, and pressing 0 • 5 • would also be treated as 0 • 5, as if the second decimal point is ignored. The ISMP user interface however would treat any number with more than one decimal point as an error. Moreover, the ISMP interface blocks the user until they correct the error; that is, for the limited class of error that ISMP recognizes, q is effectively 1; thus, in the ISMP design, q is therefore measuring the user's ability to detect non-ISMP errors. Evidently ISMP errors account for about half of all input errors and, as these results show, it is surprising that enforcing ISMP guidelines is not routine in healthcare user interfaces. Put another way, if the system can detect a class of error, the user does not need to be trained and vigilant to detect those errors.

We note that the ISMP guidelines say:

> 'They [writing numbers in the specified error-prone ways] should NEVER [original emphasis] be used when communicating medical information. This includes internal communications, telephone/verbal prescriptions, computer-generated labels, labels for drug storage bins, medication administration records, as well as pharmacy and prescriber computer order entry screens.'

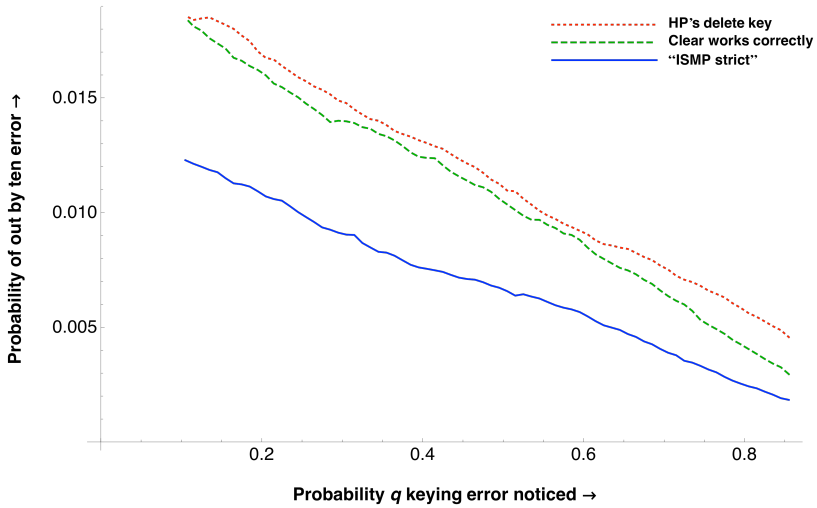

Fig. 1. Comparing three user interface designs for safety. The lower the line in the graph, the lower the probability of serious error, hence the better. The graph shows that a commercially-available but buggy user interface is least safe (of the three designs compared) and that a normal user interface is about twice as bad as an innovative user interface that implements the ISMP guidelines as part of its design rather than relying on user training for conformance to the guidelines. (The lines are based on Monte Carlo simulations of user behaviour, randomly entering typical drug doses, and therefore the plots are not smooth.)

Although ISMP do not explicitly mention medical devices like infusion pumps or linear accelerators, the intention is clearly there. Moreover, the number format guidelines we implemented in the 'ISMP strict' user interface are also recognised by the US Joint Commission National Patient Safety Goals, which specify that these error-prone abbreviations must be listed in organizational do-not-use lists.

Despite these high profile recommendations, designers have failed to make the obvious connection to user interface design! As our analysis shows, applying the recommendations would, unsurprisingly, improve safety. Figure 2 shows the effective increase in q for a single user; the worse the user was at detecting errors, the better the improvement.

Notably, the 'ISMP strict' user interface differs from the other two only in its programming: the keyboard and screen design are not changed (they are not even part of the analytic model). Any medical device or system relying on firmware or software could be improved next time the software is updated … for virtually no cost, since the firmware of these things is being updated all the time to introduce new features and fix bugs.

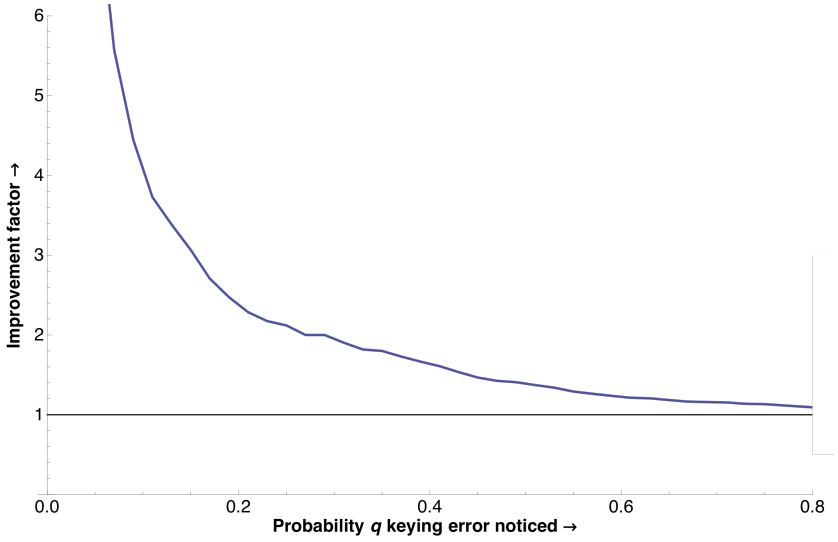

Fig. 2. Effective improvement in error detection. Effectively, the ISMP style user interface makes nurses less error-prone, and the worse the nurse, the more impressive the factor of improvement. Obviously, for a very good nurse error detection probability (q tending to 1) the improvement factor must diminish to 1 – and it never makes it worse. Specifically, the graph compares the out by ten error rate of a normal user interface (not the buggy Hewlett Packard interface also shown in Figure 1) and the improved ISMP style of user interface. (The graph above plots $s^{-1}(ismp(q))/q$ where $s(q)$ is the probability of an out by ten error for a standard user interface, and $ismp(q)$ is the probability of an out by ten error for the proposed ISMP style user interface. The graph is derived from the Figure 1 Monte Carlo simulations and therefore is not completely smooth.)

8 Conclusions

Healthcare is recognized as unsafe, and despite increasing computing interventions its safety record is not improving as well as other areas, such as civilian aviation. In fact, because computers are ubiquitous, from implants to patient record systems, healthcare is coming to be seen as a major IT problem – arguably, improving computer systems will improve healthcare outcomes better than by conventional (pharmaceutical, medical, social, etc.) interventions.

This paper explored cultural reasons why there is a lower priority to fix healthcare safety than to fix the security of financial systems or the safety of aviation systems. However, a key point is that this state of affairs need not be accepted. We showed it is straightforward to make some illustrative aspects of healthcare safer by simple engineering interventions. Essentially, the graph in Figure 1 shows that better engineered systems are safer, as one would expect. It also shows that a large improvement in dependability can be achieved by enforcing already-known rec-

ommendations to improve safety; the relative improvement is shown in Figure 2: the worse the problem the more the engineering approach helps. Essentially, the conventional approach is to treat ISMP guidance as *human* guidance; here we have shown that the ISMP guidance can be engineered into the *system* so the system can help the user notice more errors, specifically those mistakes the ISMP guidance (based on evidence) forbids.

A strong point of the present paper is that it shows that improvements in healthcare can be achieved by better engineering, and this requires no training or improved human factors awareness or even, in many cases, any changes in standard operating procedures. In other words, improving computer systems is very cost-effective and of lasting value.

The final question is how do we make improvements start to happen?

Arguably the bizarre behaviour of the Hewlett Packard calculator happened because nobody prioritised assessing it rigorously for dependability; the calculator is not a market failure, however, and apparently, no users are aware of its design problems either. The techniques used to compare designs (illustrated in the Figures) are examples of engineering assessments of safety that could be developed into rigorous safety procedures to inform design, procurement or consumer ratings. Once measured, we then need to make the safety assessments of systems visible at the point of sale and point of use. Doing so will generate economic pressure (and public pressure) to improve the engineering (Thimbleby 2013).

Acknowledgments The work reported in this paper was partly funded by EPSRC Grant EP/G059063/1. Ross Anderson made insightful comments for which the author is grateful.

References

Blythe J, Koppel R, Smith SW (2013) Circumvention of security: Good users do bad things, IEEE Security & Privacy 11(5):80–83

Cauchi A, Curzon P, Gimblett A et al (2012) Safer '5-key' number entry user interfaces using differential formal analysis. In: Proceedings BCS Conference on HCI, Vol XXVI pp 29–38. Oxford University Press

Donaldson L, Philip P (2004) Patient safety – a global priority. Bulletin of the World Health Organization 82(12):892–893

Eurocontrol (2013) From Safety I to Safety II: a white paper. European Organisation for the Safety of Air Navigation. http://www.eurocontrol.int/sites/default/files/content/documents/nm/safety/safety_whitepaper_sept_2013-web.pdf. Accessed 7 November 2013

ISMP (2007) ISMP's list of error-prone abbreviations, symbols, and dose designations. Institute for Safe Medication Practices. http://www.ismp.org/Tools/errorproneabbreviations.pdf. Accessed 7 November 2013

ISO (2012) ISO14971 Medical devices – Application of risk management to medical devices

Landauer, TK (1996) The trouble with computers: usefulness, usability and productivity. MIT Press

Pronovost P (2010) Safe patients, smart hospitals: how one doctor's checklist can help us change health care from the inside out. Hudson Street Press

Thimbleby H (2013) Improving safety in medical devices and systems. In: Proceedings IEEE International Conference on Healthcare Informatics (IEEE ICHI), pp1–13

Thimbleby H, Cairns P (2010) Reducing number entry errors: solving a widespread, serious problem. Journal Royal Society Interface 7(51):1429–1439