2013 IEEE International Conference on Healthcare Informatics

Improving safety in medical devices and systems

Harold Thimbleby
College of Science
University of Swansea, Wales
Email: harold@thimbleby.net

***Abstract*—We need to improve healthcare technologies — electronic patient records, medical devices — by reducing use error and, in particular, unnoticed errors, since unnoticed errors cannot be managed by clinicians to reduce patient harm. Every system we have examined has multiple opportunities for safer design, suggesting a *safety scoring system*.**

Making safety scores visible will enable all stakeholders (regulators, procurers, clinicians, incident investigators, journalists, and of course patients) to be more informed, and hence put pressure on manufacturers to improve design safety. In the longer run, safety scores will need to evolve, both to accommodate manufacturers improving device safety and to accommodate insights from further research in design-induced error.

I. Introduction

Approximately 11% of patients in UK hospitals suffer adverse events, of these half are preventable, and about a third lead to moderate or greater disability or death [1]. Medication errors seem to be one of the most preventable forms of error: more than 17% of medication errors involve miscalculation of doses, incorrect expression of units or incorrect administration rates [2]. The Institute of Healthcare Improvement's "global trigger tool" suggests adverse events may be ten times higher [3]. These figures come from research in different countries with different methodologies and assumptions, and suffer from a lack of reliable information [4], but there is general agreement that preventable mortality is numerically comparable to road accident fatality rates [5].

It is tempting, but wrong, to automatically blame the hospital staff [6]. One would imagine that computers, whether embedded in devices or in Health IT systems, would be part of the solution. Unlike calls to improve training or other human processes, if we can improve technology, then everybody can benefit. Yet mortality rates may double when computerized patient record systems are introduced [7]. Healthcare is now widely recongized as turning into an IT problem [8]; computers make every industry more efficient, except healthcare [9]. We clearly need informed, well-founded principles and strategies to improve safety.

Focusing on errors, however, whether leading to fatalities or not, is a mistake. Indeed, the notion of error in healthcare is a complex concept [10]: in this paper — where we are explicitly interested in improving medical devices and systems by design, and primarily engineering design at that — we take the view that we should focus on patient harm, not on error, since some errors do not lead to patient harm, some errors can be mitigtated, and "exactly the same errors" using a device may or may not lead to significant harm depending on the wider clinical context (as a simple example, an erroneous overdose of an antidote to poisoning may be fortuitously preferable to "correct" dosing if the estimate of poison is incorrect, or if the patient's response to the posion is unusually sensistive, or if an overdose is benign compared to any effect of the poison). It follows that *unnoticed* errors are the key problem, and one that good design should address.

In pharmaceuticals, it is recognized that unanticipated side-effects are unavoidable and often unexpected, so it is obvious that a rigorous process of translation and regulation, as well as open scientific scrutiny, is required before promising laboratory products are approved for clinical use [11]. So-called "use error" should be considered the unavoidable and expected side-effect of interactive healthcare technologies, particularly complex computer-based systems such as IT systems, infusion pumps, linear accelerators and so forth. In other words, we need a rigorous approach to developing technological interventions.

This paper provides a brief overview of some representative use errors, their causes, consequences and the culture in which they lie, then the paper moves to a theoretical and practical plan of action. We argue that a new approach can, over time, reduce patient harm and improve the experience of clinicians, but we need new ways to overcome deep cultural barriers to improvement. Although I have a personal bias — I want better interaction design during product development (my team's research, e.g., [12]–[14] suggests ways of achieving significant reductions in patient harm) — the suggestions in this paper do not presuppose any particular methodology to achieve the improvements that are required, merely that we make safety visible. Details will be presented later.

Note. Throughout this paper I refer to "use error" and not "user error." As Reason's Swiss Cheese model makes clear [15] — and as illustrated in our examples below — the user is only a small part of a larger system, and in fact only exceptionally is a single user *the* root cause of any adverse incident. Users are rarely even the final defence against harm; medical devices are usually the last line of defence. Just because users, like nurses and other clinicians, might be the most salient part of an incident (they are the final agent with freewill, and who made a choice that in hindsight might seem erroneous), that does not mean our language should implicitly burden them with the prejudice of user error. The error appears to become visible during use, so call it use error — but in fact the appearance is a symptom of the latent errors, the incoherence between the user and the system's design and the

978-0-7695-5089-3/13 $26.00 © 2013 IEEE
DOI 10.1109/ICHI.2013.91

appropriate use (or not) of other information such as patient records and vital signs.

II. Examples of design-induced error

It is a common design defect for a system (whether a PC or handheld device) to ignore use errors and turn them into "valid" actions that almost certainly are not what the user could have possibly intended or reasonably wanted.

A. Two radiotherapy examples

Radiotherapy involves complex calculations. Originally radiographers at the UK North Staffordshire Royal Infirmary made manual adjustments to dose, but they continued to make them after their computer system was modified to make the adjustments itself. Consequently, from 1982 to 1991 just under 1,000 patients undergoing radiotherapy received under-doses. At the Beatson Oncology Centre (BOC) in Glasgow, unfortunately the reverse happened in 2005. The Varis 7 software was upgraded and the implications were not reviewed: the original paper forms for performing calculations continued to be used. A single patient, Lisa Norris, was over-dosed in a series of 19 treatments all based on the same calculation. The report [16] says:

> "Changing to the new Varis 7 introduced a specific feature that if selected by the treatment planner, changed the nature of the data in the Eclipse treatment Plan Report relative to that in similar reports prior to the May 2005 upgrade ... the outcome was that the figure entered on the planning form for one of the critical treatment delivery parameters was significantly higher than the figure that should have been used. ... the error was not identified in the checking process ... the setting used for each of the first 19 treatments [of Lisa Norris] was therefore too high ..." [6–10, p.ii]
>
> "It should be noted that at no point in the investigation was it deemed necessary to discuss the incident with the suppliers of the equipment [Varis 7, Eclipse and RTChart] since there was no suggestion that these products contributed to the error." [2.7, p.2]

This appears to be saying that whatever a computer system does, it is not to be blamed for error provided it did not malfunction: the revised Varis 7 had a feature that contributed to an error, but the feature *was selected by the operator*. Indeed, the report dismisses examining the design of the Varis 7 (or why an active piece of medical equipment needs a software upgrade) and instead concentrates on the management, supervision and competence of the operator who made "the critical error" [10.4, p.43]. It appears that nobody evaluated the design of the new Varis 7 [6.21, p.24], nor the effect of the changes to its design, despite an internal memorandum some months earlier querying unclear control of purchased software [6.22, p.24].

Sadly, although immediately surviving the overdose Lisa Norris died. The report [16] was published just after her death.

B. Throwing away user keystrokes (I)

In these two radiotherapy examples (North Staffordshire Royal Infirmary and Beatson Oncology Centre, above), computer programs have contributed to adverse incidents, but the reports on the incidents are hard to interpret reliably, in a large part because the computer programs are above suspicion, and in the second case do not record user actions, which would help establish what users actually did, rather than what the devices they control did.

In contrast, the case of Grete Fossbakk is much clearer. Fossbakk's case does not involve a clinical event, but a financial loss that was, at the last moment, averted. In 2008, Grete Fossbakk transferred 500,000 kroner to her daughter using the web interface to her Union Bank of Northern Norway account [17]. Unfortunately, she admits, she miss-keyed her daughter's bank account number and a repeated 5 in the middle of the account number made it too long. The Union Bank of Northern Norway's web site then silently truncated the erroneous number, and this new number (which was *not* the number Fossbakk had keyed) happened to match an existing account number. The person who received the unexpected 500,000 kr spent it. Only on the steps to the court room did the bank relent and refund Fossbakk. Had she being using a clinical system, the consequence might have been an incorrect patient number rather than an incorrect bank account number.

One moral of this story is that, despite it being well known that users make typing errors, the bank did nothing to detect the error — the web site simply discarded the excess characters; nor did they record the user's keystrokes, and therefore when they asked Fossbakk to prove they had made an error, they knew she would be unable to prove the error was caused or exacerbated by their design choices. Indeed, she had (after the web site ignored her error) compounded the problem by confirming the wrong number (by clicking OK or equivalent).

In 2013, some five years later, I looked at my own Lloyd's Bank web site for transferring money to another account. It uses basic HTML to discard all but the first 8 characters of an account number. It will therefore ignore the same type of error Grete Fossbakk made and, again as it has no keystroke logging, a user would be hard-pressed to prove the bank made (or compounded) any error.

In both cases — in 2008 and now — the way a user interface is designed ignores errors that can easily be detected by the computer; worse, they are programmed to turn such errors into potentially serious unintentional errors (i.e., unintended bank account numbers), that may then not be detected by the user. In my Lloyd's Bank web site, there is *explicit* program code to ignore excess characters in bank account numbers.

Yet programming a web site to detect erroneous key presses and to log them in case of dispute is trivial. It seems that banks do not know interactive system design principles, or they do not know about the Fossbakk case, or they find the additional costs of programming properly excessive compared to their liabilities, or possibly, if they understand the issues, they do not

have the competence to program properly. Any combination is possible, though all possibilities imply the bank does not care sufficiently to employ sufficiently competent people to do a thorough job. Possibly the courts, too, are ignorant about programming. If a bank does not collect keystroke logs I think it ought to undermine their defence, as it suggests they *planned* not to collect material facts that might help uncover the truth and perhaps help their customers. One wonders whether an airplane manufacturer who refused to put black boxes in their planes would be allowed to continue to operate.

Keystroke errors like Fossbakk's happen frequently [17], and better programming could detect, avoid or reduce the chances of them turning into adverse outcomes. Moreover, for whatever reasons, the programmers involved seem content to leave things so that the user takes on the liabilities for errors. Medical systems are no different in these respects: we have found medical devices do not record adequate logs [18], they ignore errors [14], [19], [20], and so on.

Many errors with medical devices are hard to notice: small errors may have trivial consequences; consequences of errors may not be visible for some minutes or hours; users may be unwilling to report large errors; and patients may have co-morbidities so it may be hard to attribute ill-health or death to a specific cause — for all these reasons, there should be a high level of care in design. Yet many devices do not provide a key click sound (or the sound can be turned off), so a missed key press or a double key press (e.g., a key bounce) may go unnoticed.

C. Throwing away user keystrokes (II)

Handheld calculators are used throughout healthcare, from drug dose calculations, burns resuscitation, to radiotherapy, and so forth. Calculators are an easier example to discuss than many more complex medical devices, say, infusion pumps, since their purpose is well-defined, *to do arithmetic dependably*.

Many infusion pumps include calculators, though the examples we will show here are just about number entry and display, problems they share with all known number-dependent systems. We've reviewed calculator problems elsewhere [14], [20], [21] so two brief examples will serve for the present paper (in this section and the next).

I live in Wales, and I am interested in what proportion of the world's population is welsh. I therefore use a calculator to find out $3,063,500 \div 6,973,738,433$, which is the population of Wales divided by the population of the world (being numbers I got off the web so they must be right). I obtain the following results (ignoring least significant digits):

Casio HS-8V	0.04 ...
Apple iPhone portrait	0.004 ...
Apple iPhone landscape	0.0004 ...

These are all market-leading products, yet *none* of these calculators reports an error — only the last is correct. Whatever is going on inside the Apple iPhone, it could clearly report an error since it provides two different answers even if it doesn't know which one is right!

The first elecronic calculators appeared in the 1960s, and Hewlett-Packard's first calculator was made in 1968. We are no longer constrained by technology, and we've had some fifty years to get their designs right; it is hard to understand why calculators used in healthcare are not more dependable.

D. Unreliable error correction

Users make mistakes, and a common approach is to provide a delete key, or perhaps an undo key. The idea is familiar from desktop systems, but devices tend to implement these error-correcting functions in broken ways. *If the user corrects an error, the correction process needs to be reliable*. We will show that even "high quality" calculators behave in bizarre ways, and this can only encourage further use errors. Indeed, the length and intricacy of this section of the paper to describe a (completely unnecessary) bug suggests how confusing a user in a clinical environment would find the problem.

On the Hewlett-Packard EasyCalc 100, the delete key behaves in a peculiar way. If the user keys [1] [0] [0] [DEL] they will get 10; the [DEL] deletes keystrokes; yet if they key [0] [•] [•] [DEL] [5] they might be expecting 0.5 but they will get 5, ten times out. The [DEL] simply ignores decimal points — instead, it only deletes the digits. In fact, it does not even do that reliably; if the user enters a number with more than 12 digits, the calculator shows E because it cannot handle any more digits — this is a large number of digits, so this behavior is not only reasonable, but welcome (it is not present on the Casio and Apple calculators discussed above). Yet if the user continues to key digits, the [DEL] key does not delete those digits, but the twelth digit pressed. In other words, when too many digits have been pressed, the calculator *ignores* what the user is doing, so [DEL] deletes the last key the calculator paid attention to. Unfortunately, pressing [DEL] corrects the error of entering too many digits, so if the user keyed 1234567890123456[DEL] the result would be 12345678901 not 123456789012345.

That confusing example is easier to understand by writing a to mean 11 digit keystrokes and b to mean 2345 (or in fact, any sequence of digits); thus keying ab[6][DEL] is treated as a, that is ignoring all of the user's keystrokes b. One might have expected pressing [DEL] to delete the most recent keystroke, in this case [6], hence correcting the user's keypresses to ab — which is how [DEL] works everywhere else in the rest of the world. Once we see the problem, of course there is an explanation: when the calculator's display is full, the calculator ignores keystrokes, but perversely [DEL] works regardless — but because the calculator has silently ignored keystrokes, it deletes *older* keystrokes.

It is not unreasonable that the calculator's display is limited; in fact 12 digits is generous as things go (the LifeCare 4100 LED displays only 4 digits, dropping any fifth digit [22]; this is explained in the user manual, so presumably is a deliberate design choice). There are various sensible solutions:

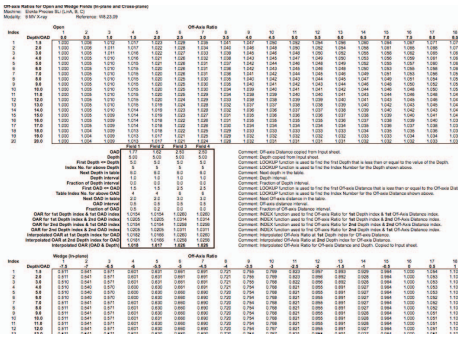

Fig. 1. Illustrating the complexity of just a small part of a routine radiotherapy calculation undertaken in an Excel spreadsheet.

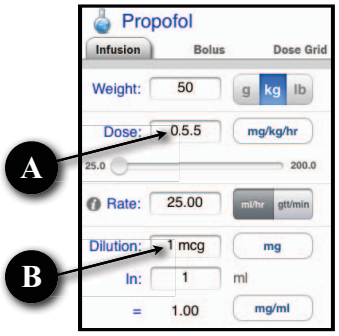

Fig. 2. Screenshot of the Infuse app by Medical Technology Solutions. Highlighted (A) is the syntax error 0.5.5, and (B) where the user has entered "1 mcg" (1 microgram) into a field that is shown in units of mg (milligrams). No errors are reported; the 0.5.5 is apparently treated as 0.5 and the 1 mcg is apparently treated as 1 mg, a factor of 1,000 out from what the user typed. The number parser is behaving as if it is stopping (and not reporting an error) when it reads a non-digit or a second decimal point; this is an elementary bug.

the simplest is that the calculator displays ERROR and locks-up when the display is full — since there has been an error, the user should notice the error and, recognising it as such, have a chance to correct it, say by pressing [AC]; the calculator could also count how many digits are keyed and how many deletes, and unlock when digits − deletes ≤ 12.

Interestingly the [DEL] on another Hewlett-Packard calculator, the 20S, behaves quite differently: [2] [•] [DEL] [5] will be 25 (the EasyCalc would get 5), and [2] [•] [•] [DEL] [5] would be 5 rather than 2.5.

In short, the key provided to help correct errors behaves like a conventional delete key much of the time, but not consistently, and differently on different models even from the same manufacturer. An inconsistent error correction function has to be worse than a consistent one, and one that varies across the same manufacturer's models has to be even worse.

Similar correction features are provided on some medical devices (and of course calculators are used widely in healthcare); unlike most calculators, though, medical devices often keep a log of user actions. It worrying to think that a user could make a keying slip, correct it, but the device would record and act on an unrelated number — because the error-correction function is badly designed. In one of the examples explained above, a nurse might have entered 0.5 (say, milliliters of morphine), but because of the broken implementation of error-correction, the infusion pump could deliver 5 mL and record 5 mL in its log — providing misleading evidence that the nurse incorrectly entered 5.

These design problems must seem trivial to most people, including the manufacturers. One might try to dismiss calculator problems by arguing that such errors are very unlikely, but one should recall that in healthcare there are numerous calculations. As [14] makes clear, similar problems beset spreadsheet user interfaces — it is a design problem, not a calculator problem *per se*; figure 1 illustrates part of a very large spreadsheet of a routine radiotherapy calculation, where there are numerous opportunities for error just for a single patient calculation. Or perhaps one would try to argue that users should be skilled — but that is made unnecessarily hard by the idiosyncratic diversity of bad designs. One might argue that the cost of getting designs right (or reducing risk to be as low as reasonably practical) is prohibitively expensive. Then one has to wonder why the earlier HP20S is safer than the later EasyCalc.

Our next example should dispel any remaining sense of triviality.

E. Kimberly Hiatt

Kimberly Hiatt made an out-by-ten calculation error for a dose of CaCl (calcium chloride). The patient, a baby, subsequently died, though it is not obvious (because the baby was already very ill) whether the overdose contributed to the death. Hiatt reported the error and was escorted from the hospital; she subsequently committed suicide — she was the "second victim" of the incident [23]. Notably, after her death the Nursing Commission terminated their investigation into the incident, so we will never know exactly how the calculation error occurred [24]. It is possible that Hiatt made a simple keying mistake on a calculator, such as pressing a decimal point twice, resulting in an incorrect number (see figure 2), or maybe she used a calculator with a broken delete key, as described above. We have elsewhere criticized medical devices that treat repeated decimal points in bizarre ways [19], though this is perhaps the least of their problems [14], [21].

F. Standards compliance

The ISO Standard 62366, *Medical devices — Application of usability engineering to medical devices*, requires an iterative development process. Devices should be evaluated and improved before marketing. The root cause analysis of the death of Denise Melanson [20], [25] included a two-hour evaluation of five nurses using the Abbott infusion pump involved in the incident. This simple study shows that the Abbott infusion pump has specific design problems. One wonders why the infusion pump was not tested in a similar way during its development, as recommended by ISO 62366.

The ISO Standard 60878, *Graphical symbols for electrical equipment in medical practice*, provides a range of standardized symbols for medical device features, such as alarms and on/off buttons. The recommended symbol for a battery is

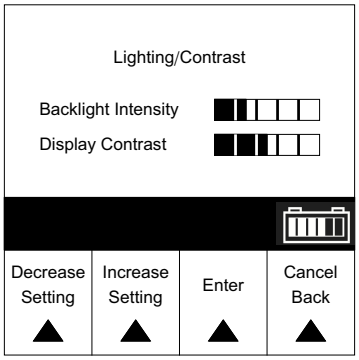

Fig. 3. A diagram drawn from the Hospira Plum A+ display panel. Is the battery 1/3 or 2/3 charged? Note that the horizontal bars for backlight intensity and display contrast have increasing black bars from the left, whereas the battery inconsistently has black bars increasing from the right. It is not clear, then, whether the battery's white sectors are indicating 2/3 charged or the black sectors are indicating 1/3 charged. (Note that the increase/decrease buttons are both triangles pointing up — neither left nor right.)

Fig. 4. ISO Standard 60878 symbol for a battery. Contrast with the symbol shown in figure 3. The standard also makes clear that the size of the darkened area is used to indicate charge. Arguably the standard might be improved if the battery was stood on end, so the darkened area was "filling" the battery against gravity in the usual way of filling a vessel.

shown in figure 4. The symbol used on the Hospira Plum A+ is shown in figure 3. It is not clear what the battery symbol chosen means, and there appears to be no advantage in using a non-standard symbol, especially one where the design is ambiguous. We have elsewhere estimated that infusion pump problems caused by dicharged batteries amount to 4% of all problems [18]; it is important to be able to read battery charge levels reliably.

The Hospira Plum A+ infusion pump was drawn to my attention through a Medical Device Alert [26] issued by the Medicines and Healthcare products Regulatory Agency (MHRA) and widely distributed. The Plum A+ has a volume control for its alarm, and the MHRA alert says there is a risk of staff failing to hear the alarm because the operation of the volume control is not consistent with the operator's manual: on some pumps, the knob is turned clockwise to increase the volume, on others, it is anticlockwise. Wiring it up or programming it inconsistently is one design problem, but more seriously the volume control knob is in a recess on the rear of the device. Since the notions of clockwise and anticlockwise are reversed on the rear of a device, and to turn the knob the operator's hand must be twisted to face back to the user, it is likely that inconsistencies are the least of the user's problems! If the sound level is critical to safe operation (it must be, else there would not have been a formal alert followed by remedial action) why is the safety-critical design feature on the rear of the device where its operation is intrinsically confusing, and anyway a user would be unable to visually confirm what level it was set to? The manual clearly shows the knob has no pointer [27], and hence it is impossible to discern the alarm sound level from it.

G. Zimed Syringe Driver

The Zimed AD Syringe Driver is, according to its web site [28], "a ground-breaking new infusion device that takes usability and patient safety to new levels. … Simple to operate for professionals, patients, parents, or helpers. With colored, on-screen guidance at each step of infusion" and a "continous improvement programme ensures that the AD Syringe Driver continually matches and exceeds the market[']s needs." In our team's paper [12] we showed that this device permits use error that it ignores, potentially allowing very large numerical errors. One particular cause for concern is over-run errors: a nurse entering a number such as 0.1 will move the cursor right to enter the least significant digits of the intended number, but an over-run (in this case, an excess number of move-right key presses) may move the cursor to the most significant digit. Thus an attempt to enter 0.1 could enter 1000.0, just through one excess keystroke. To what extent are the manufacturer's claims of "usability" based on objective measurements? (For instance, as required by relevant ISO Standards, such as 14971, *Medical devices — Application of risk management to medical devices*, which refer to the usability standards 9241, 62366, etc, discussed elsewhere in the present paper.)

H. Hospira Symbiq

The FDA Safety Information and Adverse Event Reporting Program records many design issues. Take the Hospira Symbiq Infusion System's "Class 1 Recall – May Not Respond to Selection, which affects the Symbiq type Infusers" [29]: due to software issues, the Symbiq's touchscreen may not respond to user selection, may experience a delayed response or may register a different value from the value selected by the user. A Class 1 Recall is "the most serious type of recall and involves situations in which there is a reasonable probability that use of these products will cause serious adverse health consequences or death." Yet the Symbiq had won a design award based on "systematic evaluation of three design criteria — functional obviousness, ease of operation, and creativity — and three development process criteria — user research during concept development, user research during the design process, and the use of appropriate evaluation methods" [30]. Clearly these prize criteria did not include safety.

I. Assumptions and bad logging

A morphine overdose led to the respiratory arrest of a patient [31]. A nurse the paper suggests entered a morphine concentration of 0.5 mg per mL instead of 5 mg per mL into the infusion pump; this is what the device logged. With the device thus initialized with a concentration ten times too low, it pumped a volume of the drug ten times too high. Since the pump log showed that the concentration used was incorrect, the paper [31] assumed that the nurse who operated the infusion pump was at fault. However the nurse may have performed legitimate actions that the infusion pump handled incorrectly (as explained in [32]); if so the pump design caused the error, not the nurse, or possibly some more complex interaction between the nurse and the pump (such as a design

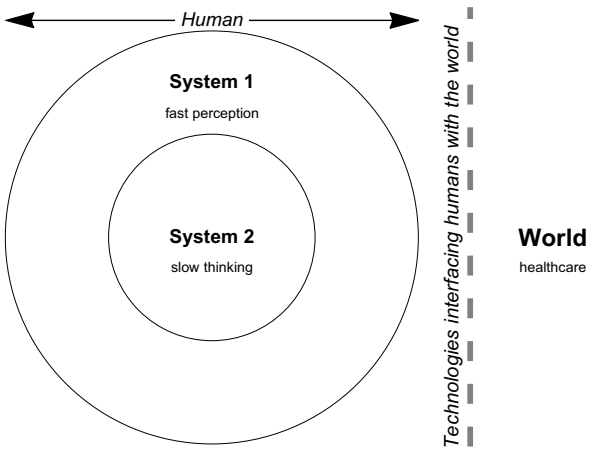

Fig. 5. Kahneman's two cognitive systems [33] illustrate how conscious thinking (System 2) can only know about the world through what it perceives using System 1. System 1 is fast and effortless, if not completely reliable; System 2 is conscious and powerful, but requires effort. Many errors occur when System 2 delegates decisions to System 1, e.g., in so called attribute subsistution [33]. Consequently, skill acquisition risks introducing sources of error if high-level defences do not compensate for the "human factors" of System 1. As sugegsted in this paper, introducing well-designed technologies to mediate the world (e.g., of healthcare) with System 1 can convert cues that System 1 would normally miss or not alert System 2 with.

misunderstanding or the hospital having many different pump designs, etc) are other candidates for the root cause of the incident. It is disappointing that the paper does not discuss the user interface of the pump.

J. Summary

Poor design and ignoring detectable error seems to be ubiquitous. The examples above were not picked out because they were egregious — some of the systems discussed are market leaders — they were picked out because they are representative. Indeed, the list of examples could be continued almost indefinitely, and could cover everything from implants to national health IT systems.

III. From problems to solutions

The common thread in all the cases above is that errors go unnoticed, and then, because they are unnoticed, they are unmanaged and then lead to loss or harm. In the case of Lisa Norris [16], modifications to a design that induce user error not only go unnoticed but are dismissed by investigators. Simply, if a user does not notice an error, they cannot think about it and avoid its consequences (except by chance). The broader consequences of this seeming truism are explored in detail by Kahneman [33], and illustrated in figures 5 and 6.

All of the example cases above are that a user makes some sort of slip, mistake, intentional error (or even that they violate the rules and bystanders are unaware or complicit in this), and they (or their team) are unaware of this error until it is too late to avoid or mitigate the consequences. In Grete Fossbakk's case, she was unaware at two levels: unaware of her error, she confirmed it. An important role of computers (beyond automation, data capture, etc) is therefore:

A. *Reduce the probability of an error occurring* — for example by reducing confusion over modes;
B. *Help notice error* — for example by reporting errors so they are noticed (e.g., sounding an alarm);
C. *Block errors* — for example by lock-out, requiring confirmation or restarting;
D. *Enable recovery from errors* — for example by providing an UNDO key;
E. *Limit or help recover from harm that has occurred* — for example by providing full details of what happened; and
F. *Other solutions*, which we will not consider in this short paper — for example a *weakest link* that is a preferential error, typically making errors visible (so they can be recovered from) but avoiding or reducing patient harm. Another important solution is to facilitate *learning* and *feedback*: if you can't solve a problem now, how can you do better in the future? — here, for example, logging user actions will help incident investigations and hence more reliable learning after any adverse event.

This list may be compared to Vincent [10], who only covers points A (but with probability 0; i.e., to prevent error in the first place), B and D. It will be important to find ways to reason about error to ensure that design strategies are (ideally) sound and complete, and lead to designs that do not interact in unmanageable ways with other systems or lead to counter-productive workarounds, and can be improved with experience.

The philosophy here is that errors *per se* do not matter; what matters is harm. Therefore unnoticed errors are a particular problem, and computers should endeavour to help reduce, mitigate, detect, block or recover from errors. We will expand further on the options below, but first some more general comments are in order.

It follows that computers (and their designers) should consider and detect actual and likely errors in order to provide this support. Interestingly, much of the focus on use error in the literature is about managing error or error recovery — for instance, providing an undo function — rather than in noticing the error, even blocking it occuring in the first place (see section *A* below).

The ISO Standard 62366 says, "Reasonably foreseeable sequences or combinations of events involving the USER INTERFACE that can result in a HAZARDOUS SITUATION associated with the MEDICAL DEVICE shall be identified. The SEVERITY of the resulting possible HARM shall be determined." (Original emphasis.) One example would be the user turning the alarm volume right down so that alarms cannot be heard — a post-market problem of the Hospira Plum A+.

The 62366 standard later discuss categories of foreseeable user action that lead to harm, namely by using Reason's error taxonomy [15] (intended/unintended errors, etc). Yet, this classic taxonomy of the error does not directly help design.

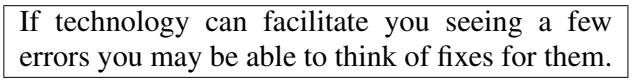

Fig. 6. How many letters "f" can you count in the sentence above? This familiar puzzle has an explantion using Kahneman's cognitive systems (figure 5). First, System 2, which performs counting, is reluctant to even answer the question because counting is "hard" and it can be rationalized away — "the answer doesn't matter" — and, besides, if I told you the answer, you probably wouldn't bother counting them to check! System 1 is delegated to perceive and find the letters "f" and then pass them over to System 2 to actually count them. Recognising the letters "f" is fluent and easy. Many people will concentrate on counting accurately and not notice whether System 1 has missed some letters: if System 1 misses or ignores some letters, there is no way that System 2 can count them, as it has no awareness of the errors (and this example is contrived to induce errors). As suggested in figure 5 technology could be used, for instance to highlight the letters "f" in the sentence, then the counting task would be both easy *and* reliable. An example technology-supported version of this problem is provided in figure 9 at the end of this paper. *Please try the problem here before reading the technology-supported problem and explanation*.

The standard is also vague how a manufacturer is supposed to implement any of its recommendations. For example, it is basic computer science to define the language the user interface implements using a formal grammar, and routine analysis of the grammar (say, by an off-the-shelf compiler-compiler) will classify sequences of events that should be properly defined — and will also define, and even implement, relevant error handling. This approach has been taught to undergraduates since the 1970s. Yet when we look at, say, the popular Graseby 3400 syringe driver, the sequence of actions [1] [•] [7] [•] [0] instead of being recognized as an error (the number keyed has two decimal points) is treated as 1.0 [19]. The device has *ignored a detectable error*, and moreover, done something bizarre with the error — turned it into a valid numeric value that is wrong. Our papers [12], [14] give many further examples.

There may be feature interaction or other unintended consequences arising through implementing these ideas. For example, detecting many errors may lead users to suffer from "alarm fatigue" and ignore warning alerts. Any systems developed should be evaluated and the designs iterated (e.g., to ISO 9241, *Ergonomics of human-system interaction*). However, just because improving systems rigorously is going to take a long time does not mean we should not start now.

A. Reducing the probability of errors

The main emphasis in the literature in human-computer interaction is on usability and user experience; arguably if usability is poor, this is easy to notice and therefore likely to get fixed (if only because of market pressure), whereas poor error management is very hard to notice, but — at least for healthcare applications — much more important. Almost all of the theory in human-computer interaction (e.g., the model human processor) assumes error-free performance! The Symbiq usability design award, mentioned above, is perhaps a case in point.

Nevertheless, there *is* a substantial and useful body of research on interaction design that has focussed on avoiding errors in the first place; notable contributions include Norman's classic works [34], [35], discussions in almost all relevant textbooks [36], etc, and specific clinically-related human factors engineering guidelines such as the John Hopkins report [37]. The ISO Standard 9241, *Ergonomics of human-system interaction*, is not only a basic standard to improve user interfaces, but has a useful bibliography. More specifically, reducing the probability of misreading numbers in the clinical context has received a lot of attention [38], yet I have not found a device that obeys all the rules for displaying (let alone entering) numbers.

It is remarkable that these basic and well-known rules, guidelines and methods are not more widely followed in the design of dependable (mission critical and safety critical) clinical user interfaces.

B. Blocking certain errors

Calculators are very bad at detecting error [20], detecting only the grossest, such as division by zero. This is very surprising since the syntax of arithmetic expressions is well understood (and allows for detecting many possible errors).

Normally errors in calculations are not obvious. Suppose we wish to calculate $\frac{1}{0.1} + 5$ but accidentally omit the 1; the relevant sequence of keystrokes is [AC] [1] [÷] [0] [•] [+] [5] [=]. We (in the calm of this paper) know that the answer should be an error, and therefore should be blocked. Indeed, on the Casio HS-8V, when [=] is pressed, the display shows ERROR 0. It might have been better if no number (0 in this case) was shown, but it shows ERROR, and the calculator is effectively locked down until the user presses [AC] to clear the error. On the other hand, the Apple iPhone calculator shows 5 when [=] is pressed, and shows no evidence of any error. In fact, the iPhone detects the 1/0 error when [+] is pressed, but the Error display is transient, probably missed by the user as they are looking for the next key to press, and certainly missed if the user proceeds — when [5] is pressed, the display of Error turns immediately to 5. The Apple calculator detects the error, but, unlike the Casio, does almost nothing to help *the user* notice the error.

Less blatant errors, such as keying too many digits (the problem exhibited in section II-C) or too many decimal points, *cannot* detected by most calculators. Worse, some calculators use an idiosyncratic concept of "too many digits" if the user has pressed [DEL] when they try to correct an error, as we saw in section II-D.

It is remarkable that the basic and well-known methods of computer science are not used to avoid such problems. These problems can be discovered by automatic processes.

C. Blocking probable errors

Blocking probable errors is a familiar feature of desktop applications. Quitting a word processor may be the correct intention or it may inadvertently lose all recent edits on a document, so the application displays a warning message (e.g., "Do you want to quit and lose changes?") and allows the user to reconsider their action. Obviously to know what a "probable

error" is requires empirical work or a design decision that the cost of an unmitigated error is so high that it should always be queried.

Some errors cannot be detected with certainty. For example, a basic infusion pump has no idea what a correct dose might be as it has no idea what the drug is. There are various techniques that attempt to reduce such dose errors, including dose error reduction system (DERS). DERS relies on an infusion pump being correctly set to the right drug, a process that in itself is error-prone, and may lead to unwarranted reliance on the DERS dosage limits.

Another approach is to notice that most infusions undertaken within a few minutes of a previous infusion are continuations. Since the patient is still alive, the previous infusion rate was not fatal; therefore signficant deviations from the previous rate should be alarmed, and the user should be asked to confirm the new value. (This might cause minor problems when a nurse wants to give a patient a bolus and temporarily increases the rate to 999 mL per hour — but such devices ought to have a bolus button!) Such a simple approach would have warned the nurses who mistakenly infused Denise Melanson with a chemotherapy dose 24 times too high, and thus saved her life.

D. Enabling recovery

Almost all desktop applications provide an undo function in case the user makes an error (which they notice and wish to recover from — if they don't notice the error, undo doesn't help!), yet virtually no device has an undo function. Many obviously dangerous devices in factories have a large red STOP button on them (often easily pressed by any body part) — and guns have locks [32] — yet medical devices rarely have buttons to immediately stop them. Even if a STOP button cannot stop everything (many errors may lead to immediate irreversible harm, and recovery as such is beyond the scope of the device involved), pressing STOP would generally simplify the error recovery, as the stopped device is no longer actively contributing to the problem.

An action that may cause irreversible harm can be delayed by the device. Sometimes the user will realize they pressed the wrong button, and a delay allows the device to undo the planned action, since it has not happened yet. Some email applications provide this feature: when the user presses SEND the email is not sent for a few seconds and, while it is still possible, a message is displayed that allows the user to have second thoughts and think whether they wish to undo the send.

E. Limit or help recover from harm that has occurred

Many approaches to limit or recover from harm are medical and beyond the scope of this paper. However if a system does not properly record what has happened, clinicians will be hampered in diagnosing the problem, and the clinicians may (after an investigation) be incorrectly blamed for doing things they did not do or did not do in the way claimed; section II-I gives an example.

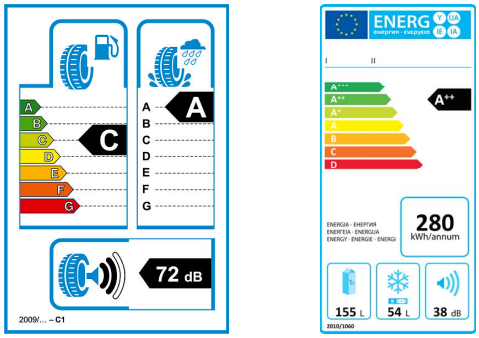

Fig. 7. Example EU product performance labels: tire label (left) and energy efficiency label for consumer goods (right). The A–G scale has A being best.

F. Summary

These illustrative examples do not exhaust the possibilities. More research is needed, combined with informed economic pressures to adopt improvements (which in turn will cost-justify the investment in the research).

IV. Making informed purchasing decisions

European Union (EU) legislation now requires car tires to indicate their stopping distance, noise level and fuel efficiency in a simple way at the point of sale; an example is shown in figure 7. This legislation follows similar schemes to indicate energy efficiency in white goods. When somebody buys something, they can now include the visualized factors into their decision making: customers, understandably, want to buy better products. Indeed, thanks to energy efficiency labelling, product efficiency has improved, in some cases so much so that the EU has extended the scales to A⋆, A⋆⋆, A⋆⋆⋆.

Interestingly, the EU has not specified *how* to make things better, they have just required the quality to be visible at the point of sale. The manufacturers, under competition, work out for themselves how to make their products more attractive to consumers. It does not matter whether the techniques are open or proprietary and give particular manufacturers commercial advantages. The point is, once safety is visible, normal commercial activity will lead to safer products.

We therefore propose a similar sheme for making the safety of medical devices visible.

The people who purchase medical devices are not the people who use them (mostly nurses), nor the people who suffer from safety problems (patients). We therefore need an adjustment to the basic scheme: the visual label of quality must not be removable. Both nurses and patients must be able to see the label on the device. If a patient questions why they are being given an infusion from a F-rated device, that will tend to raise awareness in the purchasing system to buy better devices.

V. What to measure?

The EU tire labelling scheme is not without its critics. For example, it does not indicate tire lifetime, and some manufacturers claim this is an important feature of their products. The EU has decided that only a few factors are important to visualize; it is also clear from the labelling

schemes used on consumer goods that the criteria can be revised. For example, once all tires have excellent tire noise, there is no point indicating it, and the EU could either simplify the label or target another factor to improve.

By analogy with tires, then, medical devices should show some safety ratings — ones that are critical and that can be assessed objectively — and start with these to stimulate market pressure to make improvements. Obvious issues to measure include the response of the device to use error (e.g., syntax errors during number entry) and the consequences of unnoticed over-run errors, as (for example) affect the Zimed infusion pump mentioned in section II-G.

Our simulation and laboratory work give many other examples of possible measurements [12], [13]; the John Hopkins report [37] (as just one example) implicitly suggests a wide range of important empirical measurements; and just counting how many of the ISMP rules [38], mentioned in section III-A, are adhered to would provide a simple measure directly related to safety. There is no shortage of relevant design factors to measure, then, but further work is needed to choose the critical measures that have an objective ("evidence-based") impact. It must be remembered that one does not need to get all the measures perfectly developed before being able to make an impact.

It is easy to measure tire stopping distances; we have agreement that stopping several meters before an obstacle instead of hitting it is desirable, and the use of the meter as the unit of stopping distance is uncontentious (and if you want to use feet, they are defined in terms of meters). There is no similar consensus in healthcare. Therefore any measurement process has to be combined with a process to improve, even create, the measurements and methodologies themselves. That is, we have to assess and publish the safety of a devices *and* measure the outcomes in the environments where those devices are used. In road safety, for instance, we have national figures that allow a correlation between road accidents and types of vehicle, and if we were so inclined we could also correlate down to tires, and thus we could establish more effective metrics to use for car tire safety labeling initiatives. We should do the same in healthcare.

VI. Cause for optimism?

It is very encouraging that the observations of this paper are starting to be echoed broadly across healthcare, not just as case studies but in informed reflection and expressed eloquently and effectively, e.g., [39], [40].

In the UK, about 2,000 people die on the roads annually, and cars (and tires) are subject to legally-required pre-market safety checks, and then after three years to annual safety checks, as well as random road-side checks. The safety checks are a combination of visual inspection, rigorous tests, and using advanced equipment (e.g., to measure exhaust emission quality). After a road accident, proof of roadworthiness is usually required, as it is obvious that a faulty car may cause an accident *because* it is faulty. In healthcare, we don't even have a word analogous to roadworthiness!

In the UK, about 25 people die annually from electric shock (and only about 2 while at work). Electrical installations (e.g., house wiring) and electrical appliances (e.g., toasters) are subject to required checks. Employers, additionally, have to show they ensure the safety of their workers (and visitors) by some process such as routine appliance testing, which is performed by suitably trained personnel who perform visual inspections and a few standard electrical tests. For example, an appliance that has a high earth leakage current (e.g., a non-medical Class I device with leakage more than 3.5 mA) may be failed, and labelled with a DANGER DO NOT USE sticker so it is taken out of service.

In UK hospitals about 14,000 people die annually in hospitals from preventable errors [1], and perhaps 10% of those errors are due to calculation error. Such figures are dramatically higher than electrical fatalities. Testing medical devices for safe interaction should be routine.

The usual regulatory interventions to improve products lead to so-called "regulatory burdens," which naturally manufacturers wish to reduce. Curiously, "regulatory burden" does not stop manufacturers following some standards — the Hospira Plum A+ conforms to the ANSI/AAMI ES1-1993 Standard, *Safe current limits for electro-medical apparatus* [27] for example, despite the "burdens" such conformance obviously incurs. The difference must be that electrical safety is obviously a critical issue for everybody, whereas there is very low cultural awareness of many other sorts of safety. Making safety visible at the point of sale may be resisted in the short run, but once the market sees the advantages of safety, manufacturers will willingly compete to excel. Regulations for safety are only a burden if the customers can't see how they benefit; if safety is visible, customers will pay more for better devices — this isn't a burden but a marketing opportunity.

Purchasers are not the only market force to influence. Insurers might increase excesses and premiums when low quality devices are involved in incidents.

In short, technologies where we have a mature relationship set an optimistic note for the improvement of medical technologies. If we can start to use schemes like safety labels — which are an uncontentious idea with older technologies — then this will start to raise awareness, creating a virtuous circle.

VII. Cause for pessimism?

In 2001 one of the towering figures in computer science, Edgser Dijkstra wrote,

> "... computing's central challenge, "How not to make a mess of it," has not been met. On the contrary, most of our systems are much more complicated than can be considered healthy, and are too messy and chaotic to be used in comfort and confidence" [41].

Dijkstra laments the problem of unmanaged complexity and the fundamental problems of "entanglement," namely that when a computer system entangles with another system, such as a healthcare system, we know even less about what is going on. Dijkstra's expert view is that the problems we have

identified in this paper are at least the consequence of our ignorance; it is not just a matter of trying harder, but we do not even know what the fundamental principles of good design are.

Most common programming languages are hard to use to develop safe programs and some major programming textbooks ignore error [42]. If programmers want to manage use error, there is a huge cultural gap to leap. Almost anybody can write a program that looks like it works alright. A few lines of code can make a computer do things, but for it to do those things reliably *under all potential circumstances* is an engineering problem, typically involving concepts (e.g., invariant, verification and proof) beyond the everyday needs of most programmers — and this is before the challenges of good interaction design (e.g., detecting and managing use error) are added to the requirements, to say nothing of the complexity of healthcare requirements. Typically, badly-engineered programs that appear to work are sold, with neither adequate engineering to avoid bugs nor formative evaluation to find and help correct design defects when the systems are used. In the entire world, only a few hundred people are certified IEEE software developers! Again, there is a relevant international standard: ISO 19759, *Software Engineering — Guide to the Software Engineering Body of Knowledge*.

The Dunning-Kruger Effect [43] explains why just telling people they need to program better will not help — they aren't skilled enough to understand the problems. Don Berwick expresses the same sentiment well:

> "Commercial air travel didn't get safer by exhorting pilots to please not crash. It got safer by designing planes and air travel systems that support pilots and others to succeed in a very, very complex environment. We can do that in healthcare, too."
>
> — Don Berwick, former President and Chief Executive of the US Institute for Healthcare Improvement

Healthcare systems (whether devices or IT systems, like EPRs) are driven by "clinical needs," which in turn are conceived and promoted by clinicians. It takes a significant level of maturity to recognize the difference between the alluring nature of consumer products and their likely effectiveness in clinical practice. Consumer products, such as smart phones, tablet computers, are wonderful, but designed for individual consumers. Naturally, individuals like such devices. However, it does not follow that they are appropriate for healthcare — some arguments for using tablet computers, clouds and so on in hospitals are rhetoric like arguing for ice skating. Almost everyone finds ice skating exciting, but that is not the same as it being relevant for effectively supporting clinical tasks!

To compound the problems, most consumer devices are built by very large design and engineering teams; where are the comparably-resourced teams in healthcare? The consumer market relies on rapid churn (a continual supply of new products) to keep up with new features, whereas healthcare needs stable systems that might still be in use in decades' time. Most healthcare organisations are short of money; buying technologies that (thanks to consumer market pressures) are obsolete in a couple of years does not seem like a sound long-term investment.

The consumer industry needs consumers to want to purchase their devices, and usability sells them. It is then, unfortunately, a short step to argue against innovations for safety because they are "less usable" and therefore people will not like the ideas. For example, an extra confirmation step too increase safety will add keystrokes, and the more keystrokes there are the "less usable" a device must be. This is a misconception. In fact, the pervasive consumer market culture has diverted our attention from the *dependable success* of tasks to the *sense of flow* when an error-free task is accomplished; our natural cognitive biases tend to ignore the times we fail (we may even blame ourselves rather than the design). A useful analogy from cars is that brakes make cars slower (that's what they are for!) yet it is obvious that a car without brakes would have accidents so often that task completion rates (safely completed journies) would suffer. So, a cause for pessimism is that consumers want usability as a one-dimensional concept, at any cost, even ignoring safety.

There are many differences between consumer product design and healthcare device design. Some of the things that are very enticing about modern consumer products are mobile devices (smart phones, tablets), social applications (like Facebook and Twitter), multiplayer games, and so forth; these are all new, *constructed* activities that are highly seductive. We did not do Facebook before it was invented, and we had no idea how much fun it would be. It is no accident that Apple has more financial resources than the United States.

These prominent mass-market devices were not designed for healthcare. For instance none of them are problematic for the consumer purposes for which they were designed if they disconnect or fail, even for a day or so. But a healthcare application that had the poor dependability of a typical consumer device would be dangerous.

Of course new technologies can inspire new healthcare approaches (such as Patients Online) but it should be emphasized that one's personal enjoyment of a design does not imply it is appropriate (or even enjoyable) in a work environment, let alone in a complex healthcare environment. It is very hard to make this distinction, as our next example shows.

There is evidence that nomograms are fast and accurate ways of performing drug and other calculations [44]. In a peer-reviewed article in the *Annals of Internal Medicine* [45], Grimes disparaged nomograms because they use an obsolete technology — paper. Grimes makes numerous factual errors, including making an unfavorable comparison with digital calculators because (he claims) decimal points do not get misplaced with calculators (for opposing views, see [20], [39]). Yet he is sufficiently convinced that in a later follow-up in the same journal he suggested asking for empirical evidence to support his claims would be as useful as asking for evidence for the safety of parachutes: indeed, who would sign up to a blinded, randomized trial to defy gravity? The point is, his strong support of modern computer technology (apparently

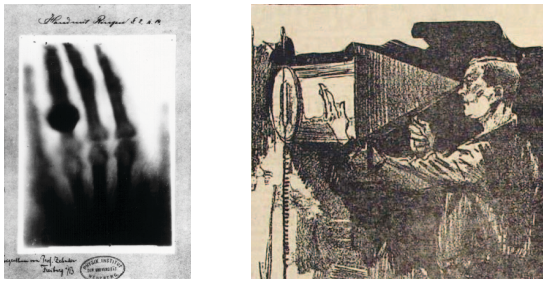

Fig. 8. (left) Wilhelm Röntgen's X-ray of his wife Anna's hand, taken in 1895; and (right) illustration of Clarence Dally X-raying his hand, from the *New York World*, August 3, 1903, page 1.

supported by the journal's reviewers and editors) overrides any objective comparisons based on safety, accuracy or speed.

It is interesting to read critiques of pharmaceutical development [11] and realize that at least in pharmaceuticals there is a consensus that scientific methods should be used, even if the science actually done has many shortcomings. In healthcare devices, such as infusion pumps, tablet computers, calculators, and so forth, there isn't even any awareness that these things do need evaluating, in the same way that proposed treatments of diseases need rigorous trials. The computerized patient record system that increased mortality [7], mentioned in the introduction, was introduced without any prior trials.

VIII. A call to action

The section on optimism (VI) is shorter than the section (VII) on pessimism!

Whether we are optimistic or pessimistic, any noticeable improvement is going to take a long time. If it is going to take a long time, we should start earlier in the process: in our teaching. If these issues are brought to the attention of our students in a constructive way (e.g., in computer science, programming and HCI courses, and in health IT courses) those students will become tomorrow's system designers, system procurers, regulators, and, whatever they become, critical users. They might also become journalists or lawyers, and work in other ways to improve the status quo.

While further work is needed, our own research has made some progress. We refer the reader to `www.chi-med.ac.uk`; here are some highlights:

- We have found that many commercial systems can be improved in ways that can be predicted using rigorous techniques (e.g., [46]–[48]). We have found ways that can be used either to evaluate designs or to help select improved designs — with potentially large gains in safety [14], [47]. We have demonstrated such techniques are effective on reverse engineered systems and can supplement the usual analyses that are performed in hindsight after an incident, such as those discussed in section II, but it would of course be preferable to use the techniques proactively during development.
- We have found that analyzing usage logs (often recorded automatically) from live hospital systems can also provide useful insights [46].

IX. Conclusions

Today's acceptance of computers (Health IT) and devices with embedded computers (infusion pumps, etc) recalls the original enthusiastic acceptance of X-rays (figure 8) — Clarence Dally, an early adopter, suffered radiation damage and later died from cancer only a few years after Röntgen's first publication [49]. When the problems of X-rays became widely recognized, their risk/benefit trade-offs were assimilated, and it is now obvious that X-rays carry risks and have to be used carefully.

Or when Ralph Nader published *Unsafe at Any Speed: The Designed-In Dangers of the American Automobile* in the 1960s [50] the car industry operated like many programmers today: they assumed "drivers have accidents" and therefore there was no obligation to improve the design of cars to make them safer. Cars from the 1960s look obviously dangerous with the benefit of our hindsight today, even to naïve eyes: they are covered in design features that are sharp and pointy — and they have no seat belts, let alone air bags.

Today's medical devices are badly programmed, and the culture is to blame users for errors — thus removing the need to closely examine design. Moreover, often manufacturers require users to sign "hold blameless" contracts [39], turning attention away from design to other possible factors, such as the hapless user. Today's papers like [7] suggest that mortality rates in hospitals can double when computerized patient record systems are introduced: even if such results are contentious, computers are not the unqualified "X-ray" blessings they are often promoted as being.

Reason's Swiss Cheese Model [15] makes it clear that the nurse (or other clinician) is never the only cause of patient harm. This paper has shown that even if the user has a "hole" (in the sense of the Swiss Cheese metaphor) the holes in the programmed devices are largely unnecessary and could have been avoided by better design. We may always have human error regardless of training and skill level but basic programming errors, particularly in simple devices like calculators and infusion pumps, are in principle avoidable — the only obstacles are culture and market forces. We have therefore proposed some simple ideas — that work well in other technology areas — to help put economic pressure on manufacturers to make their designs more appropriate for healthcare applications.

A safety labeling scheme would raise awareness of the issues and stimulate competition for safer devices and systems. It could be done voluntarily in the first place, with no regulatory burden on manufacturers. By keeping rating labels on devices for their lifetime, end users and patients would also gain increased awareness of the issues. We could start with a simple scheme and, with experience, improve it.

We need to improve. Kimberley Hiatt and many other people might still be alive if their calculators, infusion pumps, linear accelerators and so forth had been more dependable.

Acknowledgements

Partly funded by EPSRC Grant [EP/G059063/1]. Particular thanks to and appreciation of comments from my colleagues and co-workers: Ross Anderson, Ann Blandford, Abigail Cauchi, Paul Cairns, David Coyle, Anna Cox, Paul Curzon, Andy Gimblett, Michael Harrison, Tony Hoare, Daniel Jackson, Paul Lee, Paolo Masci, Patrick Oladimeji, Ben Shneiderman, Peter Szolovits, Martyn Thomas, Frankie Thompson, Brian Toft, Dave Williams.

If technology can facilitate you seeing a few errors you may be able to think of fixes for them.

Fig. 9. How many letters "f" can you count in the sentence above? We can use technology, in this case very simple typographical cues (and the answer!), to make the task much easier and more reliable. Compare with figure 6, which is the harder, error-prone, version of this problem. If you made errors in figure 6, it was probably due to System 1 not recognizing the "f" in "of" because it sounds like "ov" — which isn't an "f" sound.

REFERENCES

[1] C. Vincent, G. Neale, and M. Woloshynowych, "Adverse events in british hospitals: Preliminary retrospective record review," *British Medical Journal*, vol. 322, pp. 517–519, 2001.

[2] T. Lesar, L. Briceland, and D. S. Stein, "Factors related to errors in medication prescribing," *Journal American Medical Association*, vol. 277, pp. 312–317, 1997.

[3] D. C. Classen, R. Resar, F. Griffin, F. Federico, T. Frankel, N. Kimmel, J. C. Whittington, A. Frankel, A. Seger, and B. C. James, "'global trigger tool' shows that adverse events in hospitals may be ten times greater than previously measured," *Health Affairs*, vol. 30, no. 4, pp. 581–589, 2011.

[4] C. Vincent, P. Aylin, B. D. Franklin, S. Iskander, A. Jacklin, and K. Moorthy, "Is health care getting safer?" *British Medical Journal*, vol. 337, pp. 1205–1207, 2008.

[5] L. T. Kohn, J. M. Corrigan, and M. S. Donaldson, *To Err Is Human: Building a Safer Health System*. National Academies Press, 2000.

[6] P. Aspden, J. Wolcott, J. L. Bootman, and L. R. Cronenwett, *Preventing Medication Errors*. National Academies Press, 2007.

[7] Y. Y. Han, J. A. Carcillo, S. T. Venkataraman, R. S. Clark, R. S. Watson, T. C. Nguyen, H. Bayir, and R. A. Orr, "Unexpected increased mortality after implementation of a commercially sold computerized physician order entry system," *Pediatrics*, vol. 116, pp. 1506–1512, 2005.

[8] Institute of Medicine, *Health IT and Patient Safety*. National Academies Press, 2011.

[9] D. Goldhill, *Catastrophic Care: How American Health Care Killed My Father — and How We Can Fix it*. Knopf, 2013.

[10] C. Vincent, *Patient Safety*. Churchill Livingstone, Elsevier, 2006.

[11] B. Goldacre, *Bad Pharma: How drug companies mislead doctors and harm patients*. Fourth Estate, 2012.

[12] A. Cauchi, P. Curzon, A. Gimblett, P. Masci, and H. Thimbleby, "Safer "5-key" number entry user interfaces using differential formal analysis," in *Proceedings BCS Conference on HCI*, vol. XXVI. Oxford University Press, 2012, pp. 29–38.

[13] P. Oladimeji, H. Thimbleby, and A. Cox, "Number entry interfaces and their effects on errors and number perception," in *Proceedings IFIP Conference on Human-Computer Interaction, Interact 2011*, vol. IV, 2011, pp. 178–185.

[14] H. Thimbleby and P. Cairns, "Reducing number entry errors: Solving a widespread, serious problem," *Journal Royal Society Interface*, vol. 7, no. 51, pp. 1429–1439, 2010.

[15] J. Reason, *Human Error*. Cambridge University Press, 1990.

[16] A. M. Johnston, *Unintended overexposure of patient Lisa Norris during radiotherapy treatment at the Beatson Oncology Centre, Glasgow in January 2006*. Scottish Executive, 2006.

[17] K. A. Olsen, "The $100,000 keying error," *IEEE Computer*, vol. 41, no. 4, pp. 108–106, 2008.

[18] P. Lee, H. Thimbleby, and F. Thompson, "Analysis of infusion pump error logs and their significance for healthcare," *British Journal of Nursing*, vol. 21, no. 8, pp. S12–S22, 2012.

[19] H. Thimbleby, "Interaction walkthrough: Evaluation of safety critical interactive systems," in *Proceedings The XIII International Workshop on Design, Specification and Verification of Interactive Systems — DSVIS 2006*, vol. 4323. Springer Verlag, 2007, pp. 52–66.

[20] ——, "Ignorance of interaction programming is killing people," *ACM Interactions*, pp. 52–57, September+October, 2008.

[21] ——, "Calculators are needlessly bad," *International Journal of Human-Computer Studies*, vol. 52, no. 6, pp. 1031–1069, 2000.

[22] Abbott Laboratories, *LifeCare 4100 PCA Plus II Infuser*, Rev. 10/91.

[23] A. W. Wu, "Medical error: The second victim," *British Medical Journal*, vol. 320, 2000.

[24] Various authors, *I wish we were less patient; & To Err is Human: Medical Errors and the Consequences for Nurses*, runningahospital.blogspot.co.uk/2011/05/i-wish-we-were-less-patient.html; josephineensign.wordpress.com/2011/04/24/to-err-is-human-medical-errors-and-the-consequences-for-nurses, accessed 2013.

[25] Institute for Safe Medication Practices, ISMP, *Fluorouracil Incident Root Cause Analysis*, 2007. [Online]. Available: www.ismp-canada.org

[26] Medicines and Healthcare products Regulatory Agency, MHRA, *Medical Device Alert, MDA/2013/004*, 2013.

[27] Hospira, *System Operating Manual Plum A+*, 2005.

[28] Zimed, *AD Syringe Driver*, www.zimed.cncpt.co.uk/product/ad-syringe-driver, visited 2013.

[29] Federal Drug Administration, US FDA, *Hospira Symbiq Infusion System Touchscreen: Class 1 Recall*, www.fda.gov/Safety/MedWatch/SafetyInformation/SafetyAlertsforHumanMedicalProducts/ucm326080.htm, 2012.

[30] Human Factors and Ergonomics Society, *Two Medical Devices Win HFES Product Design Technical Group's User-Centered Product Design Award*, www.hfes.org/web/DetailNews.aspx?Id=116 (2006), visited 2013.

[31] S. Syed, J. E. Paul, M. Hueftlein, M. Kampf, and R. F. McLean, "Morphine overdose from error propagation on an acute pain service," *Canadian Journal of Anesthesia*, vol. 53, no. 6, pp. 586–590, 2006.

[32] H. Thimbleby, "Think! interactive systems need safety locks," *Journal of Computing and Information Technology*, vol. 18, no. 4, pp. 349–360, 2010.

[33] D. Kahneman, *Thinking, Fast and Slow*. Penguin, 2012.

[34] D. A. Norman, "Design rules based on analyses of human error," *Commun. ACM*, vol. 26, no. 4, pp. 254–258, Apr. 1983.

[35] ——, *The Design of Everyday Things*. Basic Books, revised edition, 2002.

[36] J. Nielsen, *Usability Engineering*. Academic Press, 1993.

[37] John Hopkins University, *Infusion Pump Workshop 2012: A Systems Engineering Approach for Human Factors Solutions*, 2012.

[38] Institute for Safe Medication Practices, ISMP, *List of Error-Prone Abbreviations, Symbols and Dose Designations*, 2013.

[39] R. Koppel and S. Gordon, Eds., *First Do Less Harm: Confronting the Inconvenient Problems of Patient Safety*. Cornell University Press, 2012.

[40] B.-T. Karsh, M. B. Weinger, P. A. Abbott, and R. L. Wears, "Health information technology: Fallacies and sober realities," *Journal of the American Medical Information Association*, vol. 17, no. 6, pp. 617–623, 2010.

[41] E. W. Dijkstra, "The end of computing science?" *Communications of the ACM*, vol. 44, no. 3, p. 92, Mar. 2001. [Online]. Available: http://doi.acm.org/10.1145/365181.365217

[42] H. Thimbleby, "Heedless programming: Ignoring detectable error is a widespread hazard," *Software — Practice & Experience*, vol. 42, no. 11, pp. 1393–1407, 2012.

[43] J. Kruger and D. Dunning, "Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments," *Journal of Personality and Social Psychology*, vol. 77, no. 6, pp. 1121–1134, 1999.

[44] D. Williams and H. Thimbleby, "Using nomograms to reduce harm from clinical calculations," in *IEEE International Conference on Healthcare Informatics*, in press, 2013.

[45] D. A. Grimes, "The nomogram epidemic: Resurgence of a medical relic," *Annals of Internal Medicine*, vol. 149, no. 4, pp. 273–275, 2008.

[46] H. Thimbleby, P. Lee, and F. Thompson, "Analysis of infusion pump error logs and their significance for healthcare," *British Journal of Nursing*, vol. 21, no. 8, pp. S12–S22, 2012.

[47] H. Thimbleby, A. Cauchi, A. Gimblett, P. Curzon, and P. Masci, "Safer "5-key" number entry user interfaces using differential formal analysis," in *Proceedings BCS Conference on HCI*, vol. XXVI. Oxford University Press, 2012, pp. 29–38.

[48] A. Gimblett, R. Rukšėnas, P. Oladimeji, A. Cauchi, Y. Li, P. Curzon, and P. Masci, "On formalising interactive number entry on infusion pumps," *Electronic Communications of the EASST*, vol. 45, pp. 1/14–14/14, 2011. [Online]. Available: http://journal.ub.tu-berlin.de/eceasst/article/view/654

[49] W. C. Röntgen, "On a new kind of rays (translated into English)," *British Journal of Radiology*, no. 4, pp. 32–33, 1931.

[50] R. Nader, *Unsafe at Any Speed: The Designed-In Dangers of the American Automobile*. Pocket Books, 1966.