Guidelines for number entry interface design (infusion devices)

EPSRC Programme Grant EP/G059063/1

Executive Summary

In most situations, errors made when entering numbers are harmless. Examples include accidentally dialing the wrong phone number or pressing the wrong button in a lift. In healthcare, the situation is different. Taking the use of infusion pumps as an example, there have been several cases of number entry error resulting in harm or death. This document aims to support a reduction number entry error though consideration of the way in which infusion pump input mechanisms are specified. It presents general principles such as implementing checks relating to validity and consistency. It describes three types of input mechanism. These include five-key, chevron and keypad interfaces. It outlines variants and describes the advantages and disadvantages associated with the various schemes.

For previous versions please contact Chris Vincent +44 (0)20 7679 0694

1. Introduction

Entering numbers into electronic equipment is a commonplace task that is happening everywhere, all the time. Despite the apparent simplicity, seemingly benign errors can become threatening in certain contexts. For example, reports relating to stockbroking, aviation and healthcare all describe serious consequences resulting from mistaken number entry [1-3]. Potential deviations include “fat finger” errors (where the wrong key is unintentionally pushed), or other “keying errors” such as mistaken substitution, deletion, repetition, or transposition. These types of error can result in order of magnitude differences when (for example) a decimal place is omitted.

Within healthcare, number entry is often mundane and unremarkable [4]. Where errors do happen, they are usually caught and corrected by the user. Taking the example of programming an infusion device, one scenario would involve healthcare professionals entering values relating rate, time and/or volume to be administered. Despite the routine and ease of doing this, if undetected, errors have the potential to harm or kill a patient [5].

As in many safety critical domains, the design of equipment can be used to reduce the likelihood of errors occurring and ease of error recovery. Studies involving infusion pumps have illustrated potential redesign that would reduce error rate [6, 7]. Current systems therefore contain features designed to prevent error. For example, there are systems that can perform checks to determine if an inputted value falls within a clinically acceptable range. Systems can block input that is obviously wrong or alert users of potential discrepancies. An example of this type of technology is the “Dose Error Reduction System” (and variants thereof) that has been widely adopted over recent years. Unfortunately, implementation has not been without issue. Without investing in the standardisation of drug stock, development of an appropriate drug library, provision of networking and monitoring of use, hospitals may not realise safety benefits [8]. This is because, for example, incomplete drug libraries will force clinicians to bypass built in safeguards, therefore circumventing benefits [9].

Whilst acknowledging the need for system wide analysis and consideration of wider socio-technical factors, when it comes to design, there are broadly applicable, generic solutions that can be used across multiple contexts. Tried and tested solutions are of proven worth in many industries, for example standard keypad layouts on phones. In this document we present a series of design guidelines, outlining principles that can be applied, to reduce the likelihood of error happening during number entry. We survey a number of alternative data entry mechanisms. Some of these solutions may suit some situations more than others. We therefore articulate the

inherent trade-offs, in order for those involved in design to optimise the extent to which a given solution matches a particular context.

2. General Guidelines

2.1 Block errors and alert users to any such blocked errors

Number entry interfaces can be designed to block errors and alert users to any such blocked errors. Out by ten errors are where the ratio of the intended number to the final processed number, or vice versa, is equivalent to, or greater than 10. In a demonstration involving multiple variants of a keypad style interface, Thimbleby and Cairns [6] suggest that number entry interfaces should be designed to block input that may result in this type of error. They present a method for halving the probability of out by ten errors by blocking the entry of numbers that do not conform to the Institute for Safe Medical Practices (ISMP) guidelines.

ISMP guidelines aim to disseminate practical, proven, ways of preventing error and relate to many areas of clinical practice. The work by Thimbleby and Cairns is based upon a subset of the lines contained within one set of recommendations: “ISMP’s List of Error-Prone Abbreviations, Symbols, and Dose Designations.” These resulted from a joint FDA / ISMP campaign to eliminate the use of error prone abbreviations [10]. Although not explicitly linked to infusion pump design, it was recommended that the list applied “wherever medical information was being communicated.”

There are three principles, originally suggested by the ISMP that have been selected by Thimbleby and Cairns:

- “Do not use trailing zeros for doses expressed in whole numbers” (use 1 mg, do not use 1.0 mg).

- “Use zero before a decimal point when the dose is less than a whole unit” (use 0.5 mg, do not use .5 mg).

- “Use commas for dosing units at or above 1,000, or use words such as 100 ‘thousand’ or 1 ‘million’ to improve readability”.

Thimbleby and Cairns demonstrate the implementation of an error-blocking interface based upon these principles, with the option of adding other relevant constraints (such as blocking numbers that are too long). They demonstrate the extent to which the interface design reduces a given type of error, using analytical methods. The authors highlight the benefit of handling number input consistently (for example ensuring that the principles are maintained across all interactions).

2.2 Perform standard checks relating to the validity of the input

Following on from section 2.1, limits can be put on time, rate and the Volume To Be Infused (VTBI) so that only legal/sensible numbers are possible. This can occur as the user is entering a value, Based upon an example using a keypad interface, a specification is provided by Thimbleby and Cairns [6]:

“Create a display that shows the character string that is in the buffer, which is initially empty. On each keystroke the character is appended to the buffer. If the user presses delete then the last character (if any) is deleted. If the user presses clear then the buffer is cleared. When the buffer is updated, it is parsed to check for validity. If the number is valid then the “enter” button is enabled or enabled and highlighted. If the number is not valid and not a prefix of a valid number then entry is blocked then the device beeps and the display turns red. If the number is a prefix of a valid number then the user is prompted to continue using a “continue” symbol”.

2.3 If time allows, get the user to enter more than the minimum number of required values

Imagine a non-urgent scenario where the user is programming an infusion pump that takes a rate, time and VTBI in order to specify an infusion. The user enters the VTBI - specified in millilitres (mL) and the rate - specified in millilitres per hour (mL/hr). Many devices are designed to work out the third value - the length of time the infusion should run for, and display this for the user to check. It is trivial for a device to calculate this value.

We recommend that in cases where commencing an infusion is not urgent, the device requires active rather than passive checking. This device would require the user to enter more than the necessary number of values (in this case three). It would then indicate any significant discrepancies. This would be like colleagues independently performing calculations and then checking the results with each other. Research has shown the potential of this scheme [11].

In the case of the user making an error, we suggest that, where possible, the system is designed to support the user in locating that error. This could be done by comparing the numbers that were entered, taking into account the order in which they were entered. In many cases it would be possible to diagnose the type of error (for example a miskeying slip versus a calculation error) and show the user how to remedy it. In a few cases this would not be possible and so a visual inspection and confirmation of the entered parameters would be required. An example of where it would not be possible to indicate an error would relate to the substitution of rate and time when entering a VTBI of 10mL, rate of 10mL/hr and time of 1 hour.

2.4 Identify patterns of user input and then apply them to optimize the interface design

Based upon the analysis of infusion pump logs, sourced across multiple clinical contexts, research has shown that some numeric keypad digits are used more frequently than others [12]. The following applies:

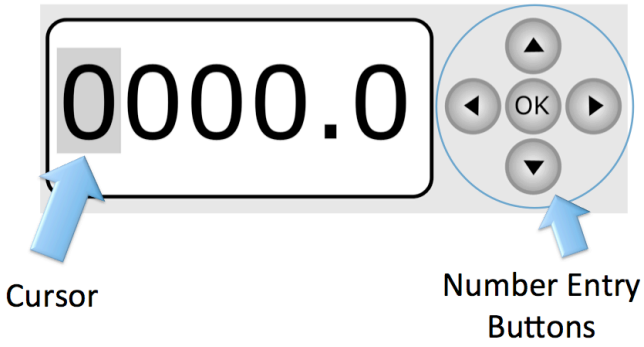
0 is a very frequently used digit, so entering a string of zeros should not require any more than one key press per digit, unless recommendations in section 2.1 or 2.2 apply.

There is also a tendency for users to want to enter a maximum value, either in terms of a rate of delivery or volume of delivery. For the models of pump investigated, this is manifested in frequent use of the digit sequence 999. Developers should be mindful of the fact that in certain clinical contexts, getting a device to deliver a solution quickly (e.g. emergency transfusion or forced diuresis), or for an indeterminate period of time (e.g. continuous infusions) may be important. There may also be the need to prime or titrate quickly. Designs that allow a maximum value to be specified using the minimum number of button presses are desirable. In these cases, consideration of the potential negative consequences of the user accidentally specifying a maximum value, or mixing up (for example) the need for a maximum rate versus a maximum VTBI should occur. Appropriate (design) mitigations should be included.

In the hospital where the logs were collected, the use of standard bag sizes and/or prescribing practices made certain digit sequences more common than others (1000, 100 and 50 were used in nearly half of all infusions). Manufacturers should consider the potential for minimizing the number of key presses required to specify these values, or provide a function to allow the immediate selection of commonly used digit sequences. In these cases, consideration of the potential negative consequences of the user accidentally specifying the wrong value should also occur. Appropriate (design) mitigations should be included.

3. Five-Key Interfaces

Five-key interfaces have a display showing the current value with a cursor highlighting the digit being manipulated (the *active* digit) and 4 buttons (left, right, up, down). The left/right buttons move the cursor between the columns, and the up/down buttons increment/decrement the active digit.



3.5 Advantages

Five-key interfaces take up comparatively less device real estate than (for example) numeric keypads. They can be implemented with fewer buttons and allow for interaction other than number entry (for example navigation of menu structures). Mechanisms of this type require the user to look at the screen while specifying a value.

3.6 Disadvantages

Five-key interfaces are likely to require a greater number of key-presses during the entry of a number than (for example) a numeric keypad. They may not be familiar to the user. The specification of the mechanism is subtle and complex. If the function assigned to the keys is overloaded with another (for example providing an undo function) then there may be situations where the consequence of pressing a key is not apparent / predictable from the perspective of the user.

3.7 Consider the impact of horizontal and vertical wraparound

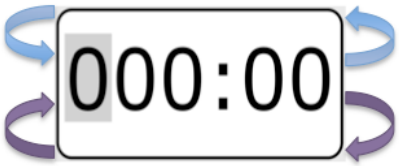
Five-key interfaces can be implemented in different ways and the variants are described below. It is important to consider how this interface is specified because subtle differences in design have a significant impact on the entered value following an identical series of key-presses. For example, when entering a volume of 950mL using the key sequence of L U U U U U L D, one implementation would result in the desired value of 950mL, another variation would result in 0.1mL. This is because in the former case, wrap around of the active digit is allowed, but independence between digits has been established (the wrap around doesn't influence the other digits). In the other case, wrap around of the active digit is allowed, however the other digits are arithmetically linked to the active digit. In this case, the mechanism reverts to a minimum value when carrying occurs.

Wraparound is the property of a numeric entry mechanism that specifies what happens when the user attempts to pass a potential limit. Examples include pressing left when the active cursor is on the leftmost digit (horizontal wraparound) or up when the active digit is reading a 9 (vertical wraparound). Horizontal wraparound concerns the effect of the left/right keys when the cursor is in the leftmost/rightmost digit (respectively). If horizontal wraparound is active, pressing left with the cursor on the leftmost digit moves the cursor to the rightmost column (and vice versa); if horizontal wraparound is inactive, this doesn't happen, and the key press is ignored or blocked.

3.7.1. Horizontal wraparound

Horizontal wraparound can speed up getting to the correct digit. In particular, getting from the rightmost digit to the leftmost in a single move can speed up entry of large values.

The drawback of this feature is that if the wrong digit is manipulated, the erroneous value can be of a large magnitude. Say, if we start from the display 0001.0 and try to go to 000.1, if we erroneously move the active digit across an extra column and then specify an increment, the result would be 1000.0 --- a much larger value than intended.



3.7.2. Vertical wraparound

Vertical wraparound concerns the effect of the up/down keys when the active digit is 9 or 0 (respectively). If vertical wraparound is active, pressing up with an active digit of 9 changes the digit to 0 (and vice versa: down on 0 changes it to 9). If vertical wraparound is not active, this

doesn't happen, and the key press is ignored or blocked. There are then additional variants that do, or do not modify the numbers in the other columns dependent on arithmetic logic.

If vertical wraparound is active, a further variant is possible: specification of arithmetic behaviour, where a vertical wraparound also results in a 'carry' between columns. In this case, pressing up on a 9 will change it to 0 *and* add one to the column to the left. Similarly, pressing down on a 0 will change it to 9 and subtract one from the column to the left.

In either case, carries could be repeated: up on 099 goes to 100, and down on 100 goes back to 099 (carrying twice in each case).

Arithmetic logic needs careful treatment around minimum and maximum values: does pressing up on 999 go to 000, or do nothing? (And vice versa.)

In addition, as per the example in the beginning of section 3.7, assuming that the user is specifying a value from left to right, pressing down on an active digit that specifies 0, has the potential to cause a subtraction that is greater than the currently specified value. In this case, an accidental extra down key press would require the user to re-specify multiple digits.

3.7.3. Alerting Blocked Actions

When wraparound is disabled, and the user attempts to pass a limit, then blocking occurs. There is a question as to: should this action be classed as an error and the user alerted in a similar way to section 2.1? It is likely that blocking the error and alerting the user will reduce the likelihood of subsequent errors. Consideration should therefore be given to alerting the user that a blocking action has occurred.

3.7.4. Initial cursor position

There is an additional question: should the active cursor be initially presented on the left hand side of the display or the right hand side? Placing the cursor on the right hand side has the benefit that the user starts with the specification of values of comparatively less magnitude, and

so accidental initiation would have less severe consequences. The downside is that some users may find this way of entering a number unfamiliar, as many systems (for example calculators) start with the specification of the digit of greatest magnitude. If this is the case, then the user has the option of immediately navigating from the right most digit to the left most digit of the required number and then commencing entry. The downside of doing this is that some may see these key presses as unnecessary and there is potential to accidentally commence on the wrong column (for example hundreds rather than thousands). This assumes Western conventions that involve writing from left to right.

Placing the cursor on the left hand side has the benefit that specification of many numbers will require comparatively fewer key presses if they involve a string of terminating zeros (assuming the mechanism, “fills in” zeros in the digits to the left). Starting number entry from the leftmost digit also reduces the potential for errors to occur [13].

3.7.5. Use of memory / undo

For arithmetic schemes, there is a question as to how the input mechanism should deal with an accidental button press that moves the value to an unintended maximum or minimum value. The accidental use of a clear function may also result in “lost” input that the user would like to recover. At this point, the concept of an undo function may be appropriate. The downside of such a scheme is that without prior knowledge of the behaviour underpinning it, a user may be unaware of its existence and/or surprised when a previously specified value arises that (for example) is not in keeping with the arithmetic logic of the input mechanism. In this way, from the users point of view, the input mechanism would not be predictable, the ultimate goal of interactive systems being to maintain predictability and reduce non-determinism [14]. For this reason, weighing the positive and negative aspects of “undo”, during the design phases is necessary. The benefit of the scheme is that it may encourage the user to explore the behaviour of the device as well as act as a recovery function. On one hand, encouraging an explorative approach to interaction behaviour allows users to try out courses of action and re-trace step if necessary. On the other hand, it may encourage risky interaction behaviour that could compromise safety.

3.8 Consider the task and specify the interface accordingly

As with other schemes, there is a trade-off between speed and safety; for example, being able to go from 0000 to 9999 in a single key press (down) might be desirable in some cases (e.g. in surgery, where an unknown but high VTBI is required, and the infusion is to be monitored). In other cases, the use of a single “down” key press to specify a maximum value may be seen as

unusual or increasing the potential for harm to occur. For this reason, weighing up the positive and negative aspects of a specification, given the need for safety and usability is important.

3.9 Consider the results of analytical studies

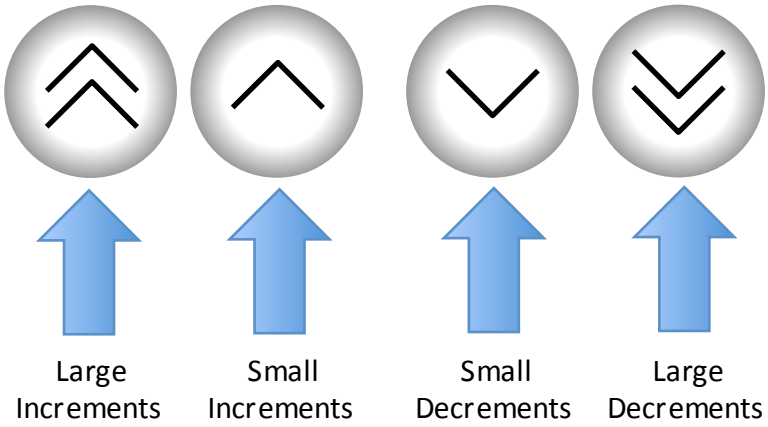
Example findings from [13] are as follows:

Wraparound (both horizontal and vertical) should be avoided because it enables errors of high magnitude(s) caused by one erroneous button press.

Having the cursor's start position on the left rather than on the right has the potential to reduce the likelihood of error.

4. Chevron Interfaces

In this guidance, chevron (arrow) style interfaces refer to number entry interfaces that offer an incremental style of number entry. A typical example relates to a system containing four buttons, consisting of: (left to right), large increments (double chevron), small increments (single chevron), small decrements (single chevron), large decrements (double chevron).



A pair of buttons is used to increase the value displayed and a second pair is used to decrease the value displayed. For each pair of buttons, one of the buttons causes a change X times bigger than the change caused by the other button. X may be set amount, or may change dependent on the size of the value that has been specified.

Typically, pressing on the single chevron keys changes the last digit of the displayed value and clicking the double chevron keys changes the second to last digit of the displayed value.

If the user presses and holds the increment / decrement button, then the rate of change of he displayed number will change. The pros and cons of this system are as follows:

4.1 Advantages

The interface is simple, as it is based around the concepts of incrementing and decrementing a value. This is likely to be familiar, although changes in step size / speed of increment may not be familiar. The scheme requires the user to look at the screen in order to monitor the change in number. This may increase the chance of them detecting an error. The scheme is constrained, so essentially the user is selecting from a list of valid values. The scheme is also economical in terms in the number of buttons required (a minimum of two) and therefore the amount of device

real estate that is required. Many of the recommendations in section 2 are not necessary due to the fact that if implemented correctly, it is impossible to specify an invalid number.

4.2 Disadvantages

This interface is comparatively slower than other number entry interfaces - about 5x slower than the numeric keypad [13]. Users are often confused about how much the number will change if they select a button, unless this value is displayed. There is therefore the possibility that the size of increment is not apparent to the user, as in some cases the increment (or decrement) changes based upon the overall value that is displayed.

There is also the possibility, that if users are allowed to push and hold in order to “accelerate” towards the intended number, they may overshoot and/or undershoot their intended value. The responsiveness of the mechanism needs to be “fine tuned” to match the reaction time of a user, which is likely to be variable as a result of individual differences (see also section 4.4).

4.3 Give feedback on the digit that will be affected by the button presses

For example, show the user that the “double chevron” increments the digit that is one to the left of the decimal.

4.4 Consider and possibly avoid the use of automatic step multipliers

Automatic step multipliers are the mechanisms that allow the user to (for example) push and hold in order to “accelerate” towards the intended number. They are driven by internal timers can make it hard for the user to understand how the input mechanism will behave [15, 16]. Studies have shown that automatic step multipliers increase the likelihood that users overshoot/undershoot their target number [17, 18]. Alternative solutions include allowing the user to take active control of step multipliers by proving an interface with more degrees of freedom (e.g. touch screen devices, pressure sensitive controls, or spring loaded joysticks).

5. Numeric Keypad Interfaces

The term numeric keypad typically describes the section of a computer keyboard, often on the right of the keyboard, used to enter numeric values. ISO/IEC 9995-4:2009 specifies the layout of a numeric section of a computer keyboard and also defines a numeric layout associated with telephones. These are referred to as the “123” layout (phones) and the “789” layout (computer keyboards). Numeric keypads on electronic equipment do not always follow these conventions, for example, the calculator layout often resembles the “789” layout, but 0 key may vary in position. For this reason consideration needs to be applied to the layout of numeric keypad interfaces.

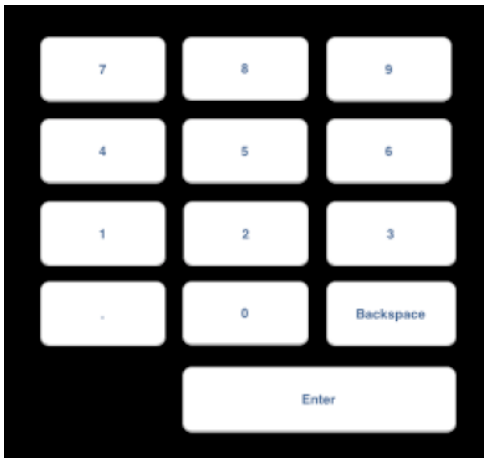
5.1 Advantages

The benefit of using numeric keypads is that they promote a familiar interaction style. They minimise the number of key presses required.

5.2 Disadvantages

The drawback of numeric keypads is that they do not encourage the user to look at the display. Users look at the keypad [17, 18]. This may lead to a higher chance of number entry error. Mechanisms of this type are also susceptible to syntax errors (e.g. entering multiple decimal points) and require the most number of buttons (typically 11 or 12). This might make them unsuitable for small or portable devices. There are also multiple ways of assigning numbers to the buttons, which can cause confusion.

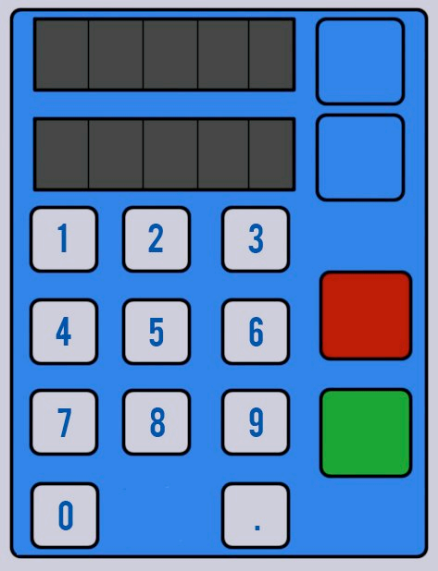
For example, in a study investigating number entry error [19], it was found that arranging the number pad with the decimal point key on the left of the 0 key led to users repeatedly typing a decimal point when they intended to press 0 and vice versa. This may be due to the reversal of the two keys, compared with layouts found on calculators and computer keyboards, which have the decimal key on the right of the 0 key.



5.3 Consider the layout

The UK guidance [20] recommends that the phone layout ("123") is used for electronic infusion devices. "The extensive use of mobile phones, and its frequent additional function as a calculator, supports a recommendation that staff are more familiar with the telephone layout." The AAMI HE75 standard [21] echoes this sentiment "For numeric keypads, the telephone-style keypad should be used unless the user needs and testing clearly indicate otherwise..." [21] p273

The current UK guidance applies an additional constraint in that the decimal place and zero buttons should not be placed next to each other and should be placed on the bottom row. Combining these recommendations and the results of experimentation [19], the following is a possible design solution:



The keys are arranged using a telephone layout. The decimal and 0 keys are at the bottom of the keypad and are non-adjacent. The 0 key is to the left of the decimal key. Place the 0 and decimal keys at the bottom of the number pad.

6. Contributors

Ann Blandford, Abigail Cauchi, Anna Cox, Andy Gimblett, Paolo Masci, Gerrit Niezen, Patrick Oladimeji, Harold Thimbleby, Chris Vincent, Sarah Wiseman

7. References

1. WSJ. *'Fat-Finger' Error Caused Oil-Stock Price Swings*. 2012 [cited 2012 6/11/12]; Available from: http://blogs.wsj.com/marketbeat/2012/09/19/fat-finger-error-caused-oil-stock-price-swings/
2. ATSB. *Take-off performance calculation and entry errors: A global perspective*. 2009 [cited 2012 6/11/12]; Available from: http://www.skybrary.aero/bookshelf/books/1748.pdf
3. FDA. *MAUDE Adverse Event Report: CARDINAL HEALTHALARIS 8100PUMP, IV, MODULE*. 2008 [cited 2012 6/11/12]; Available from: http://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfmaude/Detail.CFM?MDRFOI__ID=1235756&DEVICE_SEQUENCE_NO=2
4. Furniss, D., A. Blandford, and A. Mayer. Unremarkable errors: low-level disturbances in infusion pump use. in Proceedings of the 25th BCS Conference on Human-Computer Interaction. 2011. Newcastle-upon-Tyne, United Kingdom: British Computer Society.
5. ISMP. *Fluorouracil incident root cause analysis*. 2007 [cited 2012 7/11/12]; Available from: http://www.ismp-canada.org/download/reports/FluorouracilIncidentMay2007.pdf
6. Thimbleby, H. and P. Cairns, *Reducing number entry errors: solving a widespread, serious problem.* J R Soc Interface, 2010.
7. Lin, L., et al., *Applying human factors to the design of medical equipment: patient-controlled analgesia.* J Clin Monit Comput, 1998. **14**(4): p. 253-63.
8. Trbovich, P.L., J.A. Cafazzo, and A.C. Easty, *Implementation and optimization of smart infusion systems: Are we reaping the safety benefits?* J Healthc Qual, 2011.
9. HHFG. *Smart medication delivery systems: Infusion pumps*. 2009 [cited 2011 1/9/2011]; Available from: http://www.ehealthinnovation.org/files/SmartMedicationDeliverySystems_FullReport.pdf
10. ISMP. *ISMP's list of error-prone abbreviations, symbols, and dose designations*. 2004 [cited 2012 7/11/12]; Available from: http://www.ismp.org/tools/errorproneabbreviations.pdf
11. O' Carroll, S., *Mistakes with numbers: How interface design can influence number entry errors*, 2011, UCL: London.
12. Wiseman, S., A.L. Cox, and D.P. Brumby. A case for number entry. in Workshop: Designing and Evaluating Text Entry Methods: 30th ACM Conference on Human Factors in Computing Systems. 2012. Austin, TX: ACM.
13. Cauchi, A. Differential formal analysis: evaluating safer 5-key number entry user interface designs. in Proceedings of the 4th ACM SIGCHI Symposium on Engineering Interactive Computing Systems. 2012. Copenhagen, Denmark: ACM.
14. Harrison, M. and H. Thimbleby, *Formal methods in human-computer interaction* 1990, Cambridge: Cambridge University Press. 344.
15. Masci, P., et al. On formalising interactive number entry on infusion pumps. in Fourth International Workshop on Formal Methods for Interactive Systems (FMIS 2011). 2011. Limerick, Ireland: Electronic Communications of the EASST.
16. Masci, P., et al. *The benefits of formalising design guidelines: A case study on the predictability of drug infusion pumps*. 2012 [cited 2012 7/11/12]; Available from:

http://www.eecs.qmul.ac.uk/~masci/works/preprints/preprint-ISSE-predictability-embedded_refs.pdf
17. Oladimeji, P. Towards safer number entry in interactive medical systems. in Proceedings of the 4th ACM SIGCHI Symposium on Engineering Interactive Computing Systems. 2012. Copenhagen, Denmark: ACM.
18. Oladimeji, P., H. Thimbleby, and A. Cox. Number Entry Interfaces and Their Effects on Error Detection in Interact 2011. 2011. Lisbon, Portugal: Springer.
19. Wiseman, S., P. Cairns, and A. Cox. A taxonomy of number entry error. in Proceedings of the 25th BCS Conference on Human-Computer Interaction. 2011. Newcastle-upon-Tyne, United Kingdom: British Computer Society.
20. NPSA, *Design for patient safety: A guide to the design of electronic infusion devices*, 2010, National Patient Safety Agency: London.
21. AAMI, *HE75: Human factors engineering - Design of medical devices* 2009, AAMI: Arlington, VA.