Evaluation of Medication System Interface

James Shanteau
Kansas State University, USA
shanteau@ksu.edu

Alleene M. Pingnot, M.S., ARNP
California State University, Stanislaus, USA
apingnot@csustan.edu

James D. F. Pingnot
University of Washington, USA
pingej@u.washington.edu

ABSTRACT

Motivation - Development of a quantitative method for identification of interface design flaws. **Research approach** - Cognitive task analysis and fault analysis methodology were used to design a tool to capture expert user's experience with the interface. **Findings/Design** - Experienced users provided estimates of failure rates for each point in the fault tree. We found the need for numerous work-arounds impacted the time required for processing medications, making the interface under study inefficient. The method identified specific steps in the process of using the interface where re-design could improve efficiency. **Originality/Value** - This method can be used during system development, system selection or post deployment assessment of problematic systems. **Take away message** - This methodology provides an accessible method of quantifying error rates in specific components of the interface.

Keywords

Cognitive Task Analysis, Fault Tree Analysis, Medication Support Systems, Expert Assessment

INTRODUCTION

Medication safety has been a concern of health care providers. In 1999, the Institute of Medicine (IOM) stated that medication errors are the most common cause of iatrogenic death in the US (Kohn, Corrigan and Donaldson, Eds., 1999). Since then, increased attention has focused on this issue. The IOM identified information technology as a key tool for prevention of medication error. Physician order entry software has been found to reduce medication errors (Mekhljian, Kumar, Kuehn, et al., 2002). The health care industry is currently investing resources into purchase and implementation of computers for clinical use. Yet, poorly designed systems can contribute to problems (Ash, Berg and Coiera, 2004) and lead to resistance by end users (Neil, 2003).

There is no direct relationship between money invested in computer systems and productivity, as measured by either efficiency or error reduction (Wickens, Lee, Liu and Becker, 2004). Computers can manage large amounts of data; but to be effective for end users, systems must be efficient in terms of user time and resources and without major usability problems. There is a critical need for objective, quantifiable quality assurance measures for medical software (E. Ammenwerth & Shaw, 2005). The present study demonstrates a quantitative method of software evaluation that can identify problems in interface design.

There are two types of quality assurance evaluations for software: evaluation of internal structures and evaluation of user interface. Many quality assurance programs for software focus on internal structures of the software (Wang, Fann, & Cheung, et al., 2004; Trenner, 1995; Pizzi, Demko and Vivanco, 2001). The focus is on "the extent that software fulfills its purpose without the waste of resources" (Joshi & Misra, 1991, p. 880). Resources here refer to system memory, storage, etc., and not to human time or effort required to operate the system.

Usability testing has been an important part of software engineering for more than 10 years (Jacko & Sears, 2003). A key issue in usability involves efficiency, where efficiency is defined as enabling the user to achieve a high level of productivity with a low rate of errors (Wickens, Lee, Liu & Becker, 2004). If these criteria are met, end users will likely embrace systems. Health care professionals who use information technology report their primary hesitancy involves issues of safety and efficiency (Neil, 2003).

There are currently three common methods of evaluating the user interface: questionnaires, observations, and pilot testing (Jacko & Sears, 2003). Questionnaires and observations can be useful when questions are carefully thought out. However, users can only report their own experience to the extent that they can identify and articulate problems. Observations may also be useful, but suffer from the same lack of objectivity as questionnaires, except that the subjectivity is in the observer. Observations of activity also fail to identify cognitive issues essential to the user, such as organization of data for easy comprehension.

Scenario-based pilot studies can identify problems in software. But the value of pilot studies is limited both by how well the scenarios represent the work place. Users and evaluators may or may not be able to identify and articulate problems and their sources. These difficulties persist even when studies are carefully structured as was done in Rodriguez, Borges, Soler, et al. (2004).

The approach to interface evaluation described here collects failure estimates at each step of an interface. These estimates are used to identify points in the process that have high failure rates. Failure rates constitute quantitative measures of inefficiency in the program. Follow-up interviews of end users focus on points of failure to establish reasons for the problems and to suggest redesign.

This study is a proof-of-concept and, as such, was applied to the medication entry interface used by nurses for a particular software program. Section II describes methodology, Section III details application to a medication entry interface, Section IV describes results, and Section V discusses implications.

METHOD

Three methodologies were used to develop the interface evaluation approach described here. They are cognitive task analysis, structural fault tree analysis, and frequency estimation. Each will be described in turn.

Cognitive Task Analysis

The first step in evaluating any system is to perform a task analysis (Wickens, Lee, Liu & Becker, 2004). Task analysis allows identification of all tasks to be performed to achieve the work goal. Here, the overall goal for a nurse is to provide correct medication to patients. "Correct" in this context means that correct medication is obtained for and given to the correct patient, in the correct dosage, by the correct method of administration, at the correct time¹. Nearly all medication errors involve violation of at least one of these definitions of correctness, although many factors influence ability to prevent these errors (Page, 2004).

Various personnel in the inpatient setting may be involved in the providing medications to patients, including attending physician, nursing staff, and pharmacy personnel. However, nurses are nearly always involved in the process in several ways. Therefore, this study focuses on nursing in medication processing.

For a task analysis, the investigator must both observe the process and interview personnel. A sequential list of behaviors (tasks) is made for each category involved in the process. A task analysis describes human work behavior as compared to a system evaluation.

Fault Tree Analysis

Fault tree analysis describes the points at which errors can occur, the consequences of those errors, and their recoverability. The term "tree" refers to the many pathways involved.

Fault tree analysis reflects all system components. The rules for fault tree specification depend on the purpose of the analysis (Liu, 2001). Although cognitive task analysis and structural fault tree analysis are related, task analysis only describes the tasks that a human performs.

In the case of software, each time the program presents a drop-down menu, there is potential for branching, hence the term "tree". Another difference is that fault tree analysis goes beyond description to calculate failure rates for each part of the system. Thus, probability of failure for the system as a whole can be calculated. The strength of our method is its ability to obtain numerical failure rates. Since the present study is a demonstration, the study was limited to one part of the medication interface -- medication order entry.

Engineers designing a system usually calculate probability of failure at each step in the system (Madden & Nolan, 1999; Ferreira, Crossley, Goody & Allan, 1999). From this, steps with high probability of failure are identified. Cognitive psychology offers us a method of probability estimation that can be used to provide quantitative data for fault analysis. Instead of using historical records, our method obtains failure estimates from experienced end users.

Failure Estimates

Failures have many consequences. Catastrophic failures result in unrecoverable system errors. Although these are possible in medication systems, they are the subject of preventive action (Neil, 2003). A subtler, understudied form of failure is the need for users to repeat steps unnecessarily due to interface problems.

For example, if a medication "look up" menu is poorly designed, it may result in the user being unsure of which of several medications to select. This might result in initially selecting a similar appearing medication or a different dose. If the user recognizes this error after initial data entry and later corrects the entry, this error may not be identified as a system failure. Yet, the system then requires more time and attention of operators to insure correct data entry.

Such errors are important for two reasons. First, when users must repeat steps, this increases the time required and therefore decreases system efficiency. An inefficient system not only costs more in terms of nursing hours, it also

jeopardizes one of the measures of correct medication administration -- providing medication at the correct time. An inefficient system substitutes the error of unnecessarily delaying medication administration with other kinds of medication errors. Second, under time pressure or when distracted, these errors may go undetected with serious consequences.

There are five issues in obtaining estimates of failure: (1) quantification of estimates, (2) objectivity of data, (3) independence of estimates, (4) reliability and validity of estimates, and (5) representativeness of estimates from each participant.

Quantification.

Generally, fault tree analysis is accomplished by estimating probability of failure using historical data. However, our approach uses data from experienced users instead. Gigerenzer (1996, 1998) found participants perform better when information is presented as frequencies rather than point probabilities. He argues people naturally think in terms of frequencies of events. Therefore, we asked participants to estimate frequency rather than probability of failure.

Since probabilities are scaled on a 0-100% scale, frequencies were placed on a similar scale. This allows us to treat frequency estimates interchangeably with probability estimates for purposes of calculation. Here, the scale used was 0-10 -- a condensed version of 0-100.

The response scale was anchored to ensure that participants correctly understood the scale. We used anchors of "0 = always works" and "10 = always fails". Participants were asked to estimate frequency of failures in 10 times doing the same process.

Objectivity.

This is based on the method of data collection. First, participants provided data twice. This controls for inadvertent over- or under-estimations (Thaler, 1985). Secondly, estimates were framed differently to control for tendencies to overestimate either success or failure. The first time, participants provided estimates of success for each step; the other time, they provided estimates of failure for each step.

To prevent problems from order of data collection, counterbalancing was used. That is, each participant was administered the data collection tools in the opposite order of the participant preceding them.

Independence.

This was addressed by instructing participants to make estimates only for the step described, without regard to other steps. Independence was also addressed by having the investigator read to the participant, instead of giving the participant a list of all steps and letting them to complete them at will.

Reliability and Validity.

Reliability was addressed by having the investigator administer the tool face-to-face with each participant individually. This helped ensure participants understood each question and provided their best estimate of failure. Second, our method included two replications; thus, similarity of responses implies a reliable measure.

Validity was addressed via participant selection. Novice users are unlikely to provide valid failure data because they lack experience with the system. Further, they make mistakes as they learn any system. These mistakes are not system failures, but rather learning experiences. Nurses who use the system daily for months or years are better able to provide valid estimates.

Representativeness.

Data cleansing is an important step in analytic methods that rely on mean estimates. Outliers (unusual responses) can unduly influence the mean. Extreme data are unlikely to represent behavior of interest. They can cause the mean to be under or over estimated. Data cleansing methods identify and correct for outliers so resulting means are representative.

A handy method of examining outliers is the box plot, sometimes referred to as a skeletal box plot or a box-and-whiskers plot (Ott, 1993). Such plots provide three important measures: (1) mean, (2) upper and lower quartiles, sometimes referred to as "fences", and (3) outliers, ie, data points that lie outside the fences. The investigator can infer what is going by looking at the pattern of the box plot.

Analyses

Descriptive analyses are first conducted using standard statistical software. This provides estimates of means, standard deviations, ranges, etc. Box plots are constructed for each step in the process. If most data falls between the fences, it represents the experience of most users.

The standard method for dealing with outliers is to move the estimate to the fence. For example, if most participants provide estimates between 0-2, but one individual gives an estimate of 7, the estimate for that individual is adjusted to be the fence (ie, 2). This helps insure that results represent the population under study.

Quantitative failure data can be plotted to show system efficiency visually. Graphing makes it easy to identify points with unusually high failure rates. This highlights problems which users might not be fully aware of. Follow-up questions were asked of experienced end users to identify the nature of problems and to make suggestions for system re-design

APPLICATION

System Specification.

There are many medication-support software systems on the market. The method demonstrated here could be applied to any of these. The medication program in use at a Midwestern hospital was selected to demonstrate our method.

The system used for this study had physicians write orders on paper, which were then transcribed by nurses or pharmacists. This study focused on one system module -- entry of medication orders. Users progressed from one field on a page to another in a linear fashion. There was little choice of path and no allowance for reversing course to correct errors once the user progressed to another page. This will be discussed further under Fault Tree Analysis.

Participants.

A sample of expert end users provided estimates of frequency of failure (and success) for each step. Six registered nurses, who had been using the system for several years, volunteered to participate. These nurses entered medication orders into the system when physicians gave verbal orders, ordered medications over the telephone, or when pharmacists were not working (evenings, nights, weekends and holidays). All had been using the system daily for about three years. Institutional Review Board Permission was obtained and all human subject protections were followed

Task analysis.

The study focused on one subset of tasks in medicating inpatients. There are many kinds of medication that might be ordered and many different administration methods for medications. Since oral medication is most common, we focused on the transcription of new oral medication orders into the system. The second author conducted interviews of experienced system operators (different from those used to gather data) to develop a model of the tasks necessary to enter a medication order into the system.

The nurses' overall goal was to correctly enter an order into the medication system, often called "noting an order". Task analysis identified 13 sub-tasks in the transcription process.

Fault Tree Analysis.

The process used by the nurses was analyzed to develop a fault tree. In most fault tree analysis, there would be multiple paths to be followed. The entry of an oral medication order into the electronic chart was only one section of one branch of the overall fault tree. However, this part of the tree is enough to test our method.

Analysis of the steps required for entering an oral medication order resulted in a list of 25 steps to perform the 13 sub tasks. The "top event" (Liu, 2001, p. 274), of most importance to overall system performance, is the same as the overall goal. However, the steps in the fault tree for the system differ to some degree from those for the task analysis, reflecting system design.

Data collection.

Two interview tools were developed from the fault tree analysis. Each step in the fault tree was addressed by one question in the interview. The second author interviewed each participant individually, and recorded answers to each question. Failure was defined in terms of frequency of repeating a step before going on to the next step. Success was defined in terms of completing the step correctly the first time.

At the end of the interview, nurses were asked about overall success for a single medication order (reflecting all steps in the process). This question reflected frequency with which an order, once entered, was found to have been entered erroneously on final review.

RESULTS

Comparisons of the two questionnaires for each participant revealed quite similar results. Therefore, they were pooled for analyses. This process provided a total of twelve estimates at each step in the medication entry process (two for each participant).

Box plots of all responses to each question were used to identify outliers. Two participants had one outlier each at different steps in the process. Therefore, the outliers appeared to reflect idiosyncratic patterns of the participant rather than system problems. These two outliers were brought to the fences as discussed in the method section.

Average failure rate across all steps were 0.99. Step 5 had a mean failure rate that was nearly twice the average of the others; step 5 entailed response to an allergy screen. This was a flag designed as a safety feature. If no new allergies had been reported since the user last accessed this chart, the correct response was "B" for "By pass". Study participants noted that, because the keystroke required for a negative response (B for "Bypass") was different from the keystroke

required for similar screens (N for “No”), users frequently gave the wrong response. This problem is common in human factors and is easily remedied once recognized²⁷.

Step 20 also had a failure rate more than twice the average failure rate. Step 20 is the point at which the user “files” the new order. Because keys to be pressed for “file” and “exit” are similar, nurses reported that they would occasionally press the “exit” key when they meant to “file” an order. Depressing the “exit” key deleted all unfilled orders and, thus, the user had to begin the order entry process all over again.

This kind of error was reported to be more likely at high stress, “busy” times. Inadvertently deleting a single or a series of medication orders constitutes a serious failure in this system. This could be catastrophic when urgent orders are being processed, as in a clinical crisis.

Redesign of just these two steps would provide important increases in efficiency. Moreover, the redesign would not be that difficult.

Participants in this study reported that, although the system was clumsy to use, they were able to use “work arounds” to make the system function successfully. They reported a nearly zero rate of error to the last question in the interview, designed to identify when a medication was found to be entered erroneously once the process was completed.

Although it is possible that participants inflated their success rates, the estimates were consistent in their reports of failure (or success) rates for specific steps in the system. In addition, administrative personnel reported that medication error rates had been reduced to less than 1% by the computer system studied here. The contrast between failure at specific steps in the system vs overall failure is reflective of (1) motivation to use the system successfully and (2) ability to develop “work around” routines (Jacko & Sears, 2003, p. 41) to enable use of a poorly designed system. A “work around” is a user-devised method of accomplishing a task in spite of, rather than because of, software system. The nursing staff that participated in this study reported that total time to process orders and obtain medications had grown following system implementation.

The issue at hand is not identifying problems in a particular medication software system. Rather, we wanted to demonstrate a method for identifying design flaws in medical software systems. As can be seen here, a task analysis of the human work, followed by a structural fault tree analysis involving estimates of failure rates, provided quantitative data to objectively identify steps in need of redesign.

DISCUSSION

If the evaluation approach had relied only on total numbers of errors, this medication software system would score well. However, the reports of nurses suggest this system is quite inefficient. In health care, not only does time = money, but also time = lives saved. In emergency care, speed of treatment is the essence for saving lives. Therefore, efficiency of a system to accomplish a task should be the key measure of system success.

Our method is similar to the structured interview method used by Rodriguez, Borges, Soler, *et al.* (2004) in that reports are obtained from experienced users. However, Rodriguez *et al.* pre-identified system components they wished to evaluate and asked participants their opinion about those components. In contrast, our analysis is based on quantification of failure rates at each step to identify problem areas, rather than relying on observers to decide the location of problems in the system. End users provide greater insight into the process that impact efficiency at key failure points.

As pointed out in a recent editorial (Ammenwerth & Shaw, 2005), medication systems should be continuously evaluated. The method of system evaluation described here demonstrates that system faults can be readily identified. Our method provides a simple, yet powerful, method of system analysis. Faults thus identified can be subjected to further study to identify solutions to the problem.

On-going system monitoring would ensure any need for re-design can be identified early. This would be similar to a drug company’s post marketing reporting system. Reports generated correlating errors with higher than average failure rates for parts of the system allow re-engineering focused on the problem areas. Our method of software evaluation can also be used to compare pre-marketing prototypes or to test specific design features, such as a new menu or decision tool. Each branch of a system can be studied separately and the overall efficiency calculated. Inefficient paths or screens can be identified for further analysis and redesign.

CONCLUSION

When evaluating efficiency of a software system, it is important to take into account all aspects of its use. For example, timesavings realized for physician medication entry are unimportant if the software doubles the medication processing and dispensing time for nurses. Also, a system that makes efficient use of physician time, but fails to address the time required for pharmacists to review and dispense medications does not effectively provide medications to patients. Likewise, a system that permits easy medication order entry and retrieval, but creates difficulty in cost accounting is not an efficient system for administrative oversight. A complete evaluation of medical informatics software must include all

interfaces used by doctors, nurses, pharmacists, accountants, administrators, and auditors. Our method can be used with all interface components of a total medication system.

REFERENCES

- Ammenwerth, E. & Shaw, N.T. (2005) Bad health informatics can kill – is evaluation the answer? *Methods of Information in Medicine*, 44, 1-3.
- Ash, J.S., Berg, M., & Coiera, E. (2004) Some unintended consequences of information technology in health care: the nature of patient care information system-related errors. *Journal of the American Medical Informatics Association*, 11, 104-112.
- Ferreira, L.R.C. Crossley, P.A. Goody, J. & Allan, R.N. (1999) Reliability evaluation of substation control systems. In *IEE Proceedings-Generation, Transmission and Distribution* 146:626-632.
- Gigerenzer, G (1996) How to improve Bayesian reasoning without instruction: Frequency formats. *Psychological Review*, 102, 684-704.
- Gigerenzer, G (1998) Ecological Intelligence: An adaptation for Frequencies. In Cummins, D. & Allen, C. (Eds.) *The Evolution of Mind.*, Oxford University Press, 9-29.
- Jacko, J.A. & Sears, A. (Eds.) (2003) *The Human Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*. Mahwah, New Jersey: Lawrence Erlbaum Associates, Inc.
- Joshi, S.M. & Misra, K.B. (1991) Quantitative analysis of software quality during the “design and implementation” phase. *Microelectronics and Reliability*, 31,879-884.
- Kohn, L.T. Corrigan, J.M. Donaldson, M.S. (Eds.) (1999) *To Err Is Human: Building a Safer Health System*. Committee on Quality of Health Care in America. Institute of Medicine, Washington, D.C.: National Academy Press
- Liu, S. (2000) Verifying formal specifications using fault tree analysis. *Proceedings of the International Symposium on Principles of Software Evolution*, 2000. 272-281. IEEE.
- Madden, M.G.M. & Nolan, P.J. (1999) Monitoring and diagnosis of multiple incipient faults using fault tree induction. *IEE Proceeding.-Control Theory and Applications*, 146, 204-212. IEEE.
- Mekjhan H.S. Kumar, R.R. Kuehn, L. Bentley, T.D. Teater, P. Thomas, A. Payne, B. & Ahmad, A. (2002) Immediate benefits realized following implementation of physician order entry at an academic medical center. *Journal of the American Medical Informatics Association*, 9, 529-539.
- Neil V. (2003) Physicians force halt to Cedars-Sinai CPOE rollout. *Modern Physician*. Downloaded on 23 January, 2003: <http://www.modernphysician.com>
- Ott, R.L. (Ed.) (1993) *An introduction to statistical methods and data analysis*, 4th Edition. Duxbury Press, Belmont, California. 96-99.
- Page, A. (Ed.) (2004) *Keeping Patients Safe*, Committee on the Work Environment of Nurses and Patient Safety Board on Health Care Services, Institute of Medicine Report. Washington, D.C.: The National Academies Press.
- Pizzi, N.F. Demko, A. Vivanco, R. (2001) Discrimination of software quality in a biomedical data analysis system. *Annual Conference of the North American Fuzzy Information Processing Society-NAFIPS*, 3, 1702-1707. Vancouver. BC. Canada.
- Rodriguez, N.J. Borges, J.A. Soler, Y. Murillo, V. Sands, D.Z. (2004) A usability study of physicians’ interaction with PDA and laptop applications to access an electronic patient record system. In *Proceedings of the 17th IEEE Symposium on Computer-Based Medication Systems*, 1063-7125. IEEE Computer Society.
- Thaler R. (1985) Mental accounting and consumer choice. *Marketing Science*, 4, 199- 214.
- Trenner L. (1995) Prototyping for Usability: The benefits of peer group review. *IEE Colloquium on Integrating HCI in the Lifecycle*. 73, 5/1-5/2. IEEE. London, UK.
- Wickens, C.D. Lee, J.D. Liu, Y. Becker, S.E.G. (2004) *An Introduction to Human Factors Engineering*, 2nd Ed. Upper Saddle River, New Jersey: Prentice Hall