An effective feedback mechanism to enable Health Information Technology (HIT) users and other stakeholders to report adverse events that occur after system deployment is an essential line of defense against risks to patient safety (Billings, 1999). The goal of the Patient Safety Informatics – Cognitive Analysis Model (PSI-CAM) project is to develop a practical theory-based framework to aid adverse event analysis and to apply it to patient safety issues reported to the Informatics Patient Safety (IPS) Office of the Veterans Health Administration (VHA), Department of Veterans Affairs (VA). This framework is based on a user-centered model of human interaction with technologies (Norman, 1986) and as such it helps ensure the analysis includes consideration of the relationship between the user-technology interface and the adverse events reported. Here we describe the framework and illustrate its use with real-world examples of issues reported to the VHA IPS Office.

### INTRODUCTION

**Engineering Informatics Patient Safety**

When trying to manage patient safety risks related to how users interact with Health Information Systems (HIS), it is valuable to have defenses against adverse events at all stages of the software engineering lifecycle. Further, it is valuable to have defenses that take a user-centered approach at all those stages - including the post deployment stage where predicted use turns into actual use and the system goes through various upgrades and enhancements over the lifetime of its deployment. Figure 1 shows example defenses for four of the basic stages: Requirements Analysis, Design, Implementation, and Deployment.

User-centered approaches during requirements analysis help ensure sufficient user requirements are considered. One user-centered approach might be the development of “use cases” from a user perspective to make sure the requirements are grounded by an understanding of what type of users will use the system, what their goals will be, what they will interact with to achieve those goals, how particular parts of the system will enable them to complete tasks in pursuit of those goals, and what types of user-interaction errors need to be avoided for each task considered (Carroll, Rosson, Chin, & Koenemann, 1998). In addition to developing specific user-interaction models, another approach would be to include patient safety and usability requirements, such as "efficient to use" and "difficult to make an error" (Nielsen and Mack, 1994), in the non-functional requirements section of the Business Requirements Document (BRD).

During the design stage, user-interface style guides can be used to help develop interfaces that reduce the likelihood of errors and which are efficient to use. Such style guides are developed from general models of effective user interaction and an understanding of how to make it less likely user-interface errors will occur. Standard design inspection techniques, such as pluralistic evaluations (Bias, 1994), can take a user-centered approach by having multidisciplinary teams (e.g., involving health care providers, software engineers, patient safety analysts, and human factors experts) walk through predicted sequences of user
interaction with design mockups and discuss the patient safety and usability consequences of those interactions.

When the system is further along in the implementation process, higher fidelity evaluations can take place involving real-world users working with the system and trying to complete representative work tasks. Scenario-based usability testing (Dumas and Redish, 1999; Tullis and Albert, 2008) is one user-centered approach, and there is the option of designing scenarios to probe at potential patient safety risks for specific contexts of user interaction (Rogers, Patterson, Chapman, & Render, 2005). Performance metrics like the number of tasks successfully completed and not completed are typically collected as is the time taken to complete those tasks. Specific sequences of user action and system reaction are normally collected to help determine if intended sequences were actually followed and what sequences of actions led to errors. When an error occurs, how the user reacts to it will also normally be captured. Participants may be asked to think aloud while performing tasks to help determine why they took the actions they did at each step (Ericsson and Simon, 1993).

After a system is deployed, user interactions will be actual, not just predicted or simulated. Ethnographic techniques such as field observations (Hutchins, 1995) and semi-structured interviews (Cooke, 1994) are user-centered methodologies conducted by experts in socio-technological systems and human-technology interaction that can be used to proactively seek out unanticipated side-effects and other issues after deployment. These methodologies involve capturing and tracing information about user interactions and decision making in the workplace (Woods, 1993). However, it is also important to provide non-punitive safety reporting systems for those actually using the deployed system to do their daily work. The Aviation Safety Reporting System (ASRS), founded in 1976 and now available on the Internet at http://asrs.arc.nasa.gov, has been used as a model to create similar systems for other high risk environments. This includes health care where, for instance, the Patient Safety Reporting System (PSRS) has been developed and is also available on the Internet at http://psrs.arc.nasa.gov. As Charles Billings, one of the designers of the ASRS system has noted (Billings, 1999), if a system’s users – the people at the “sharp end” of day-to-day operations – are encouraged to report the safety problems they encounter to a program they can trust, safety goals will be reached much sooner than if we never hear the stories of those lessons learned.

It is important that information about adverse events, including near misses, is captured so that changes can be made in the system where necessary, just as changes were made during the earlier stages of development. Again, it is valuable to take a user-centered approach, meaning that it is important to capture the user interactions that contributed to the reported adverse event. Models of user interaction with information systems, such as Electronic Health Records (EHRs), have driven the development of methodologies used at the other software engineering stages to help manage patient safety risks and improve usability, and the same approach can be taken to address reports of patient safety issues in deployed systems. Information must be collected that facilitates developing an understanding of what relevant user interactions took place, what the user intentions were when known, and what the response was to errors that occurred. However, for this to occur, mechanisms must be in place to facilitate collecting that type of information and level of user-interaction detail when an incident is reported and investigated.

**PRACTICE INNOVATION**

**Analysis of Reported Patient Safety Issues**

In addition to supporting patient safety risk management at earlier stages in the software engineering lifecycle, one of the functions of the Veteran Health Administration’s (VHA’s) Informatics Patient Safety (IPS) Office is to investigate reported adverse events that are related to users interacting with HIS. In order to better understand the issue reported, assess the patient safety risk and make prioritized recommendations to resolve the causes identified, IPS conducts a Root Cause Analysis (RCA). This is an established methodology for systematically tracing what behaviors, actions, inactions or conditions led to a harmful outcome with a focus on identifying root causes rather than symptoms (Bagian et al., 2001).

When a patient safety issue is reported to IPS a Patient Safety Analyst reviews the initial report and then conducts an investigation to gather more information about what happened or almost happened (Taylor, Chapman, & Wood, 2012). This involves working with Department of Veterans Affairs (VA) system developers to better understand the software design and, where applicable, replicate the issue and document the user interactions with screen captures when possible. As already discussed, understanding cognitive factors surrounding adverse events in the use of Health Informatics (HI) is critical for detecting all contributing causes and developing effective redesign solutions. Therefore, as shown in **Figure 2**, an IPS Human Factors
expert is also consulted to provide a user-centered perspective on the reported adverse event.

**Patient Safety Issue Description**

Documented sequence of events leading to adverse event(s)

Including indicators of risk severity, frequency, and detectability

**Cognitive Perspective**

- Goals and intentions?
- Perceptual demands?
- Human information processing demands?
- Memory demands?
- Keystroke and direct manipulation requirements?
- Human-Computer Interfacing errors?
- Etc.

**Outcomes**

- Numerical Risk Analysis (to support prioritization)
- Alternative mitigation strategies (to provide flexibility)
- Human Factors perspective (to support proposed solution(s), indexing, and pattern analysis)

Figure 2. Human Factors analysis within the patient safety issue analysis process

This involves reviewing the documented issue and tracing events from the user(s) perspective while considering the cognitive challenges that were involved at each step and if a less than optimal interface design contributed to an erroneous user action or non-action.

The Human Factors expert then works with the Patient Safety Analyst and other subject matter experts relevant to the particular issue to conduct a Healthcare Failure Mode and Effects Analysis (HFMEA) (DeRosier et al., 2002) - a variation on the Failure Mode and Effects Analysis FMEA (Stamatis, 1995) - which results in an assessment of the patient safety risk. After the risk is assessed, the Human Factors expert helps identify risk mitigation strategies and, where applicable, helps redesign the interface(s).

**Error Causing "Gulfs" in User-EHR Interaction**

The IPS Office has used Don Norman’s “Seven Stages of User Activities Involved in Performance of a Task” (Norman, 1986) as a basic framework to help identify and organize a set of fundamental errors that can occur as users interface with EHRs. As Zhang et al. (2004) have noted, to be comprehensive, descriptive, predictive, and generalizable, a cognitive taxonomy should be based on a sound cognitive theory that has explanatory and predictive power, and Norman's model can provide such a theory for creating taxonomies of medical errors related to user-technology interaction in performance of a task.

**Figure 3** shows a mapping of error types to Norman’s model. Because this is a catalog of errors linked to a cognitive model to support patient safety issue analysis, we call this the Patient Safety Informatics - Cognitive Analysis Model (PSI-CAM).

When clinicians use the VHA's Computerized Patient Record System (CPRS) and other computer systems to provide patient care, they do so with a clinical goal (see "establishing goals" in **Figure 4**), such as "Diagnosing a patient’s condition" or "Developing a treatment plan." In order to help achieve such goals they form intentions such as to "Review the results of recent lab work" or "Prescribe particular medications." Clinicians translate such mental intentions into plans of physical action specific to the computer tools. For example, prescribing a medication using CPRS requires navigating to the Meds tab, creating a new order, selecting the medication, and signing the order using specific interface commands.

CPRS and other clinical software are designed to respond to the user actions in particular ways. When the software outputs information to users, they perceive that output from the system and interpret what is perceived. For instance, if the user clicks a "Submit" button during the execution stage, the user will likely expect some kind of feedback that the submission was successful. Therefore the user may look towards the screen in order to perceive that feedback and based on what is seen, the user will interpret the submission as successful or not.

In general, at this stage the user is trying to answer the questions "What is the system doing now?" and/or "What state is my request/data in now?" To answer these questions the system must present the right information, in the right place, in a form the user can understand, and at a time the user expects it.

When users have difficulties translating intentions into actions supported by the software there are what Norman calls *Gulfs of Execution*. Similarly when it is difficult to evaluate the true state of the system, or the real-world entities of interest (e.g., a description of the diagnosis or treatment plan for a patient), there is what Norman calls *Gulfs of Evaluation*. When these gaps are small and translation requirements are minimal, people think the system is easy to use and intuitive, and the likelihood of a user-interaction-based error is reduced.
Thus, when VA health care facilities began using bar-coded wristbands for patients and bar-coded labels for medications they sought to ensure the "five rights" (right patient, drug, route, time, and dose) of medication administration (Shannon, Wilson, & Stana, 1995; Johnson, Carlson, Tucker, & Willette, 2002) by making it more difficult for a "gulf in execution" to contribute to a medication administration error.

Similarly, efforts to create meaningful graphical representations in the VA's "Vitals" software have made it easier to see trends and relationships at a glance and thus reduce the likelihood of a gulf in evaluation related to how the information is presented. In designing Coversheets, also referred to as dashboards and overview displays (Tufte, 1990; Woods and Watts, 1997), for the VA's CPRS and Bar Code Medication Administration (BCMA) software, further reductions have been sought in potential "gulfs in evaluation" by making it easier to quickly obtain commonly needed information and to navigate to over-related displays (Chapman, 2004).

The mapping of standard human-computer interaction error types to particular stages of this process model (e.g., a "data entry problem" when seeking to complete a sequence of actions to select a patient's records, or an "information representation problem" when trying to interpret vital signs for a patient represented a certain way on the screen after they have been presented and visually perceived) affords systematically identifying error types while tracing the particular sequence of events over time during each Execution-Evaluation cycle of Norman's model, and thus helps reveal all the contributing cognitive factors. This is in contrast to unstructured checklists of error types that are not put in the context of a temporal model of user interaction.
**FINDINGS**

The result of applying PSI-CAM to a single reported issue is a list of standardized error types that can be tied to particular points in the sequence of events that make up the reported patient safety issue and standard human factors references for those error types. Standard solutions for those problem types can then be referenced from human factors literature and used to help propose design solutions to development teams and other stakeholders.

As more issues are reported and investigated, the IPS Office is continuously building a database of issues, thus creating a resource that can be used for evidence-based data mining of particular issue types.

**Interaction Gulfs Addressed in CPRS (v.28)**

The VA’s CPRS version 28 (v.28) was released in March 2011. From a patient safety perspective, CPRS (v.28) contains enhancements designed to address 18 specific issues that were reported to and processed by the IPS Office. The process involves collaborating closely with the VA health care facility sites reporting the issues, clinical specialists, and developers. In an analysis of changes released in CPRS (v.28), it was found that 17% (3/18) involved gulfs of execution and 89% (16/18) involved gulfs of evaluation, with most cases involving multiple categories. In this section, we demonstrate both types of gulfs and several subcategories from PSI-CAM by describing six issues we shall refer to as cases A through F.

**Issues involving Gulfs of Execution**

As previously noted, issues in the area of gulfs of execution appear as the system doesn't support the execution of correct user intentions or requires unintuitive actions by the user. For example, Case A considered the situation where in order to prevent possible errors in data entry, Emergency Department physicians entering orders before midnight, were prevented from placing unit dose orders to be given after midnight. While investigating this issue the Patient Safety Analyst found that the Inpatient Medication for Outpatient tab was designed to prevent date entry errors by blocking orders that were entered on a previous day’s date, even if the time from entry to dose administration was only a few hours. CPRS (v.28) addressed this problem by allowing dose orders to be entered for administration up to 23 hours in the future.

Another form of inefficiency can occur when invalid options are actually available to the user. This was the case for CPRS (v.27) infusion orders, as seen in Case B. Previously, for Unit Dose orders, CPRS displayed all medication routes for the dosage form of the orderable item as possible medication routes that the user could select. By not constraining the choices to only those that were valid, the system forced users to make a more complicated decision than necessary. This increases decision time and creates an unnecessary opportunity for error. For example, even if an orderable item was only to be given as intramuscular (IM), the provider was able to select a potentially inappropriate medication route of subcutaneous (SC) or intravenous (IV). Furthermore, CPRS would not display default routes for infusion orders with multiple items—even if all the orderable items on the order shared the same default medication route. This forced users to perform unnecessary actions (selecting the delivery route) even when a correct default route existed. Both of these data entry problems were corrected in CPRS (v.28) by making the user options more tailored to the state of the user interaction.

Case C is an example of a problem caused by making an option available to the user that is logical, but can’t actually always be executed because of a data retrieval synchronization problem. With previous functionality, the user could select the Patient Inquiry button and then try to open a different patient’s record before/while the initial patient’s record loaded. This allowed for an execution error in which the patient inquiry data would load for the second patient record selected instead of the first even though the Patient Inquiry button was selected before the second patient record was selected. Developers addressed this potential patient misidentification issue by modifying CPRS (v.28) to disable the Select New Patient menu item until after the Patient Inquiry information was returned and displayed.

**Issues involving Gulfs of Evaluation**

As previously discussed, designing for effective evaluation means making sure that the right information is available to the user at the right time, in the right place in the interface, and in a form that is appropriately salient and makes sense in the given context.

A key type of “evaluation” problem is when critical information exists, but is not visible to the system user when they need it, as occurred in Case D. In CPRS (v.27), when providers began ordering medications, the caption at the top of the form read “Medication Order” and did not indicate whether the patient was inpatient or outpatient. Providers believed there was a risk of incorrectly specifying the type of medication
order because the inpatient/outpatient status was too easy to miss. From a cognitive perspective, the problem was that the information necessary to make the correct choice for type of medication order was not readily apparent to the user at the time of order entry. The CPRS (v.28) solution was to add this information to the form caption, which makes the medication ordering interface more accurate, complete, and evident.

Another visibility problem occurred in Case E. With previous functionality, more than two Category (CAT) II Patient Flags could only be viewed by moving a scroll bar. The concern was that if the users did not notice the scroll bar, they would only consider the first two CAT II flags displayed and not scroll down to see the other flags, causing a potential for inappropriate or contraindicated treatment. Since the display is also alphabetical, if sites have several CAT II flags on a patient, high priority flags like “suicidal” would not be displayed without scrolling. For this reported problem, the resolution was to add a count of the number of flags so the user has a prompt to notice when there are more than two, and also a splitter bar to enable the user to increase the size of the scrollable list of flags.

The CPRS functionality “Remove Pending Notification” button (Case F) created a problem because when enabled and selected by one of many providers that receive the notification, the software was coded to delete the notifications for all other recipients, which meant the targeted provider for the notification may not see it, which could subsequently lead to a delay in treatment for the patient the notification was regarding. To help mitigate the patient safety risk involved with this problem, part of the resolution was to update the CPRS User Guide to include a warning about removing pending notifications. Unfortunately, this relies on the users’ memory of the user guide to execute the correct action, and there is no indication in the software interface itself, but a software change is targeted for a future version of CPRS.

**Strength of Intervention**

Risk mitigation recommendations apply safety principles that are based on human factors engineering reflecting best practices, such as user-centered design, and support standardized IT principles (e.g., testing and release processes). However, design changes to correct problems have to be developed while considering the level of effort involved, interdependencies in code, estimated completion times, and other competing requests. It is therefore valuable to offer multiple solutions and to rate them based on their ability to effectively address the issue.

The National Center for Patient Safety (NCPS) lists a “Recommended Hierarchy of Actions” in the Root Cause Analysis Tools flipbook used by the RCA teams nationwide at VA Medical Centers (VAMCs) for general patient safety event analysis. As shown in Figure 4, this hierarchy of actions classifies interventions as stronger, intermediate or weaker in effectiveness at reducing the likelihood of reoccurrence of the adverse events.

Stronger actions are those that eliminate or nearly eliminate the vulnerabilities by making it difficult to do the task incorrectly (e.g., a forcing a function). Intermediate actions are those which reduce the reliance of an individual’s memory or vigilance to do a task (e.g., perceptual indicators). Weaker interventions are those that require an individual to complete a task based upon their knowledge and memory without additional support (e.g., after training).

![Figure 4. Strength of Intervention Spectrum with examples (Adapted from VHA NCPS RCA Tools Cognitive Aid, 2002)](image)

According to this scale the cases previously discussed would be rated as shown in Table 1.

**Table 1. Strength of Intervention for Examples Cases**

<table>
<thead>
<tr>
<th>Case</th>
<th>Category</th>
<th>Intervention</th>
<th>Strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Execution - Data entry</td>
<td>Refined functionality to better support the data the user needs to enter</td>
<td>Stronger</td>
</tr>
<tr>
<td>B</td>
<td>Execution - Data entry</td>
<td>Refined the users options and default actions to dynamically match what is appropriate for the current state of task completion</td>
<td>Stronger</td>
</tr>
<tr>
<td>C</td>
<td>Execution - Data retrieval</td>
<td>Eliminated the possibility the user could unintentionally initiate a data retrieval synchronization problem</td>
<td>Stronger</td>
</tr>
<tr>
<td>D</td>
<td>Evaluation - Missing information</td>
<td>Now display the information that was missing</td>
<td>Stronger</td>
</tr>
<tr>
<td>E</td>
<td>Evaluation - Data visibility</td>
<td>The information is now more salient (but still requires deliberate cognitive processing)</td>
<td>Intermediate</td>
</tr>
<tr>
<td>F</td>
<td>Evaluation - Data delivery</td>
<td>Now provide information in the user guide</td>
<td>Weaker</td>
</tr>
</tbody>
</table>
DISCUSSION

A model of human-technology interaction can help guide all stages of EHR system development and maintenance. This includes post-deployment refinement from an analysis of reported issues that is not bug-centric, but user-centric. When a patient safety issue is identified, it is important that it be investigated sufficiently and with input from relevant disciplines to accurately capture what happened and why. Understanding that user interaction with the interface to information technology is cyclical in nature, and specific types of problems may occur in that cycle, can help those tracing the events over time to flush out what will often be multiple contributing factors.

The error types contained in PSI-CAM are designed to support identification of fundamental interface design issues as a user cycles between the Execution and Evaluation stages; however, this doesn’t mean other error types are not important or that other error taxonomies cannot be used in conjunction with PSI-CAM. In an abstraction hierarchy of health care interface error types the PSI-CAM errors can be regarded as relatively low-level, whereas errors such as patient identification or medication errors are higher-level examples of the PSI-CAM error types. Such higher level error types may later be explicitly included in the IPS taxonomy.

When a patient issue has been confirmed and solutions are considered, it is valuable to consider the strengths of the alternatives in terms of their ability to reduce the likelihood that the problem will reoccur. Stronger solutions will be more effective in "shrinking" the gulfs discussed in this paper, but less strong strategies can still help when they are relatively easy to implement and stronger solutions will take longer or are not practical.

Work has begun on developing a standard protocol for pattern analysis to facilitate the process of looking for patterns in the IPS patient safety issue database. Such analysis will help develop a deeper understanding of what types of problems are frequently occurring and why, as well as what types of solutions are being proposed and implemented, and the effectiveness of mitigation strategies.

ACKNOWLEDGEMENTS

We would like to acknowledge the non-human factors professionals involved in the IPS patient safety issue reporting and analysis process who are essential to the success of the system. This includes those who report patient safety issues and those who work with the IPS Office to help determine what occurred and why, and to discuss the practicality of alternative solutions. The efforts of the IPS Patient Safety Analysts should also be acknowledged as it is they who do the "heavy lifting" to collect information on the user interactions and outcomes that make application of the PSI-CAM model possible.

Roger Chapman is a Cognitive Systems Engineer for Collaborative Work Systems (CWS), Inc. under contract to the Informatics Patients Safety (IPS) Office.

The views expressed here are those of the authors and not necessarily those of the Department of Veterans Affairs.

REFERENCES


