Zheng, X., **Bolton, M. L.**[C], Daly, C., & Biltekoff, E. (2020). The development of a next-generation human reliability analysis: systems analysis for formal pharmaceutical human reliability (SAFPHR). *Reliability Engineering & System Safety*, *202*, 15 pages. https://doi.org/10.1016/j.ress.2020.106927

The Development of A Next-Generation Human Reliability Analysis: Systems Analysis for Formal Pharmaceutical Human Reliability (SAFPHR)

Xi Zheng[a], Matthew L. Bolton[a,*], Christopher Daly[b], Elliot Biltekoff[a]

[a] *University at Buffalo, State University of New York, Department of Industrial and Systems Engineering, Buffalo, NY, USA*
[b] *University at Buffalo, State University of New York, Department of Pharmacy Practice, Buffalo, NY, USA*

Abstract

Medication errors originating in community pharmacies are a serious patient safety hazard. However, due to the complexity of the community pharmacy environment, current experimental and observational studies are insufficient to address these problems. Furthermore, the static nature of traditional, model-based human reliability analyses (HRAs) are not able to handle the dynamic environmental elements that can impact human performance. To address this issue and allow analysts to accurately predict medication error rates, we develop a new HRA called the Systems Analysis for Formal Pharmaceutical Human Reliability (SAFPHR). This method addresses the limits of previous HRAs by combining concepts from the Cognitive Reliability and Error Analysis Method (CREAM) HRA with probabilistic model checking, a computational tool for automatically proving properties about complex, stochastic systems. In this paper, we use SAFPHR to analyze a common community pharmacy dispensing procedure, compare our results to published error rates, and use our results to explore interventions that could reduce error rates. We ultimately discuss our results and explore how our method could be developed in future research.

Keywords: Human reliability analysis (HRA), cognitive reliability and error analysis method (CREAM), medication errors, formal methods, probabilistic model checking.

1. Introduction

Medication errors are a serious problem. In an eight-year period of study, a Johns Hopkins team estimated that more than 250,000 people in the United States die every year from medical errors. This makes it the third leading cause of death in the US after heart disease and cancer (Makary and Daniel, 2016). Beyond their cost in human lives, medication errors also increase the economic burden on society. The center for the disease control estimates that each year medication error costs exceed 3.5 billion dollars (not including lawsuits Bates 2007). For these reasons, improving medication safety has been a patient safety goal of the joint commission for many years (Parker, 2013; The Joint Commission, 2014, 2015, 2016a,b).

There are several places where medication errors can occur. One significant location is community pharmacy. Contemporary comprehensive studies in the United States found that between 0.057% (Szeinbach et al., 2007) and 1.7% (Flynn et al., 2003) of prescriptions filled by community pharmacies have dispensing errors. The higher rates are concerning because they correspond to an average community pharmacy making approximately four medication errors a day and two clinically significant errors a week (Flynn et al., 2003; IOM, 2006). As such, medication errors originating from community pharmacies are a major threat to patient health and safety (Parker, 2013; The Joint Commission, 2014, 2015, 2016a,b).

Unfortunately, medication errors in community pharmacies are complex and difficult to understand. The work environment of community pharmacy is dynamic and multifaceted. This often results in pharmacists having heavy and ever-changing workloads. It is important to note that pharmacists need to not only perform dispensing responsibilities, but also play important roles interacting with patients, resolving medication discrepancies with providers, and managing pharmacy personnel. As a result, pharmacy errors can manifest in many places.

Medication errors are also poorly understood. While voluntary reporting systems exist, the number of medication errors is under-reported (Ashcroft et al., 2006). If the provider perceives no harm to the patient, "near-miss" errors will not be reported (Allan and Barker, 1990; Wilson et al., 1998) and it is estimated that approximately 25% of errors are undocumented and unreported (Kaushal et al., 2001; Pape, 2001). Furthermore, almost all community pharmacies are private and thus not required to share information about their procedures and errors. This, together

*This paper is an expanded version of a paper by Zheng et al. (2017) that was originally presented at the 2017 International Annual Meeting of the Human Factors and Ergonomics Society. This article describes a significantly extended modeling and analysis approach. It also presents new modeling and verification results and features a deeper discussion.

*Corresponding author
Email address: mbolton@buffalo.edu (Matthew L. Bolton)

with the fact that pharmacies use different procedures, makes it difficult to get comprehensive, consistent data.

Observational and experimental studies are sometimes used to gather error data and identify the major cause of errors (Flynn et al., 2003; Berdot et al., 2013; Lao et al., 2016). However, it is not easy to gain the necessary access to real community pharmacies due to patient privacy issues. Furthermore, no matter how well the study is designed, experimental and observational methods can be time-consuming, expensive, incomplete, and can miss conditions that could arise in an actual system. These issues suggest that community pharmacy medication errors are a prime candidate for the application of model-based approaches like human reliability analyses (HRAs).

HRAs allow analysts to estimate human error rates based on the sociotechnical factors that impact human performance (Swain and Guttmann, 1983; Hollnagel, 1998a; Liu et al., 2018; Liao et al., 2019). The most advanced HRAs work by having analysts describe procedures as sequences of tasks (Hollnagel, 1998a; Bell and Holroyd, 2009). Then, analysts compute probabilities of errors based on expert's subjective estimates about the sociotechnical conditions that influence each task. While powerful, HRAs have major limitations (Hollnagel; Fujita, 1992; Swain, 1990). They focus exclusively on human error, rather than the entire system. They are also static and thus ignore system dynamics and interaction effects between errors that can influence the distribution of sociotechnical factors and thus error rates. Attempts to apply predictive HRAs to pharmacies have failed because of these limitations (Rantanen and Deeter; Deeter and Rantanen, 2012; Rantanen et al., 2012).

We propose to address these shortcomings by creating a next-generation HRA that is based on probabilistic model checking. Probabilistic model checking (Kwiatkowska et al., 2011) allows analysts to automatically prove properties about models of complex stochastic systems. Specifically, it can definitively determine how likely different system behaviors and outcomes are, while accounting for all the different possible behaviors, interactions, and system dynamics included in a formal model. This means that probabilistic model checking could address the limitations of HRAs by capturing dynamic system changes and interactions between humans and other errors in a system.

In this paper, we create a novel, formal, proof-based approach to HRA called the Systems Analysis for Formal Pharmaceutical Human Reliability (SAFPHR$\mathbb{R}$; pronounced "safer") that is capable of capturing how medications move through a pharmacy that also includes the types of errors, the potential contributing factors, and the dynamism that impacts these factors. To do this, we build on the basic Cognitive Reliability and Error Analysis Method (CREAM) (a well-validated approach to HRA) to transform expert estimates of relevant environmental and cognitive factors into human error rates. The dynamic and environmental conditions that can influence error rates as well as non-human sources of medication errors will be accounted for through the use of probabilistic model checking. This will allow us to include all the dynamics that impact human error and mathematically prove how error-prone particular tasks are. Also, this method will support the automatic computation of error rate predictions by virtue of probabilistic model checking. SAFPHR$\mathbb{R}$ will enable us to assess the overall reliability of a pharmacy's procedures and compare them to potential alternatives and interventions.

In this article, we develop SAFPHR$\mathbb{R}$ by extending preliminary work we did to show that probabilistic model checking (Zheng et al., 2017) could be used with CREAM to make accurate predictions about pharmacy error rates. Below we provide background on HRAs, CREAM, and probabilistic model checking. We then detail the steps of implementing and applying SAFPHR$\mathbb{R}$ to analyze a full community pharmacy dispensing procedure. Finally, we compare our results against published error rates and explore new avenues of future research.

2. Background

2.1. HRAs

Human error is a critical factor in many industrial failures, where it is estimated that they contribute to between 60% and 70% of accidents (De Felice et al., 2012). Therefore, many Human Reliability Analysis (HRA) methods have been developed to assess the human contribution to errors and failures. HRAs are generally classified into two generations dependent on whether the methods were based on probabilistic risk assessment or cognitive activities, respectively.

First generation HRAs (such as Technique for Human Error Rate Prediction (THERP) (Swain and Guttmann, 1983; Swain, 1987), the Human Error Assessment and Reduction Technique (HEART) (Williams, 1986, 1988), Human Cognitive Reliability (HCR) (Hannaman et al., 1984) and the Operator Action Tree (OAT) (Wreathall, 1982)) treat humans as mechanical components and thus human errors are regarded the same as an equipment failure. In these methods, assessors decompose operator tasks into components and then account for the potential impact of performance shaping factors (PSFs) such as time pressure, equipment design, and stress (Bell and Holroyd, 2009). By combining these elements, the assessor calculate the nominal human error probability: the probability of error for a particular task. The PSFs are used to portray the positive or negative effect on performance and provide a statistical basis for modifying nominal human error probability levels (Boring et al., 2006). While useful, first-generation HRAs are often criticized for failing to consider things like the impact of context, organizational factors, and errors of commission (Hollnagel, 1998a).

Second-generation HRAs improve on these by accounting for these sociotechnical factors in their error rate predictions (Fujita and Hollnagel, 2004; Zheng et al., 2017; Reer, 2008). To accomplish this, the second-generation methods

account for interactions between human operators, production processes, the organization, and the environment and how they impact models of human cognition (Hollnagel, 1998a; Bye et al., 1999; Kim and Jung, 2003; Kim et al., 2006; Lee et al., 2011; Di Pasquale et al., 2013; Zhao and Smidts, 2019). Collectively, these developments move the focus of HRAs to the cause of errors rather than just their frequency (Di Pasquale et al., 2013). The most notable of the second generation tools are A Technique for Human Error Analysis (ATHEANA) (Cooper et al., 1996; Commission et al., 2000) and Cognitive Reliability and Error Analysis Method (CREAM) (Hollnagel, 1998b). However, ATHEANA has no formal guidance about how to compute overall error rate probabilities. Thus, researchers who use it will "borrow" a quantification framework from another method (such as THERP) (Bell and Holroyd, 2009; Xie et al., 2007). As such, CREAM is largely considered the leading second generation method. We thus use CREAM as the basis for the work presented here. Details on CREAM appear in the next section.

2.2. CREAM

CREAM (Hollnagel, 1998a) is the leading second-generation HRA (Bell and Holroyd, 2009) that posits that human performance is determined more by the situation in which a task is performed than it is by inherent properties of the task itself. As such, CREAM, in its basic form, calculates ranges of human error probabilities based on assessed values of sociotechnical factors called Common Performance Conditions (CPCs; Table 1). These were chosen so that the minimal number of CPCs could adequately describe the criteria influencing human performance (Hollnagel, 1998a).

To use basic CREAM, analysts describe procedures as sequences of tasks. Then, CPCs are assessed by an expert who, for each task, rates whether the conditions associated with each CPC improve human task performance, reduce it, or are not significant. Two of the CPCs, Goals and Time of Day, have only two levels: not significant and reduced. Four of these CPCs (Conditions, Available Time, Goals, and Collaboration) are dependent on other CPCs and are thus adjusted based on assessed CPC values (Hollnagel, 1998a). Fig. 1 describes this process.

After adjustments, the number of CPCs that are improved and the number that are reduced are counted. These counts map to one of four Contextual Control Model (COCOM) control modes, each with a range on probabilities of human error (Hollnagel, 1998a,b) (Fig. 2). *Scrambled* control describes a situation where a human loses situation awareness (due to high task demands, unfamiliar situations, or unexpected events) and actions are chosen randomly with little or no thinking involved. *Opportunistic* control corresponds to situations where the human chooses actions inefficiently due to incomplete planning or a failure to anticipate events fully. This can occur because the time to perform the task is too constrained or because the human does not clearly understand the context under which a task is performed. *Tactical* control is characterized by situations where actions are chosen through planning that is based on following known procedures or rules. However, this planning will have a limited scope and the procedures and rules will not necessarily be appropriate in all situations. Finally, *Strategic* control encapsulates situations where a human plans and chooses actions after a full consideration of the situation. In cases with little control, such as in the scrambled and opportunistic modes, the probability of making a failure is high. Conversely, when the level of control increases, the likelihood of the human making an error goes down (Hollnagel, 1998a).

Table 1: CREAM CPCs (adapted from Hollnagel 1998a)

CPC	Description
Organization	Relates to the roles and responsibilities of team members as well as the quality of additional support, communication systems, safety management systems, instructions, guidelines, and oversight.
Conditions	Relates to physical working conditions such as ambient lighting, screen glare, alarm noise, and interruptions.
Support	Relates to man-machine interfaces. This includes the information on control panels, computerized workstations, and operational support provided by decision aids.
Procedures	Relates to procedures, including operating and emergency procedures, familiar patterns of response heuristics, and routines.
Goals	Relates to the number of goals or tasks a person is required to pursue or attend to at the same time.
Available Time	Relates to the time available to carry out a task and corresponds to how well the task execution is synchronized to process dynamics.
Time of Day	Relates to the time of day; in particular, whether the person is adjusted to the current time.
Experience	Relates to the quality of operators' training and their level of operational experience.
Collaboration	Relates to the quality of the collaboration between crew.

CREAM has proved to be useful in a number of different applications including nuclear power plants (Hollnagel et al., 1999), food manufacturing (Geng et al., 2015), radiation therapy (Castiglia et al., 2008), and hospital pharmacies (Rantanen and Deeter; Deeter and Rantanen, 2012; Rantanen et al., 2012). However, it has limitations inherent to all first- and second-generation HRAs. In particular, CREAM is static. Thus, it does not consider interactions between errors or how rates will change dynamically as a system operates. There have been other attempts to develop third-generation HRAs to account for system dynamism (Kirwan et al., 2004; Bell and Holroyd, 2009; Li and Mosleh, 2019). However, the majority of these are based on first-generation HRAs (like HEART) and thus lack the theoretical and cognitive grounding of CREAM (Kirwan et al., 2004; Bell and Holroyd, 2009). All, including more modern ecologically reactive methods like those

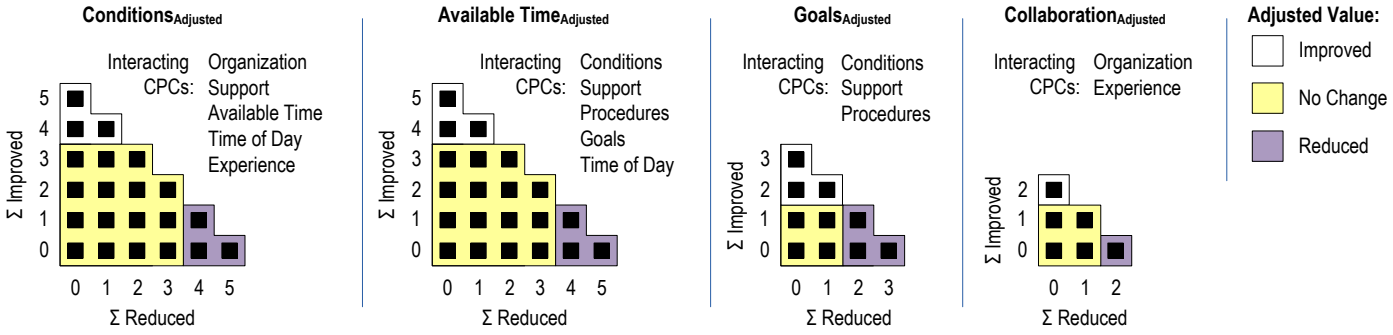

Figure 1: CREAM's method for adjusting CPC values to account for dependencies between CPCs (Hollnagel, 1998a). Each graph shows one of the four CPCs that are adjusted along with a list of the CPCs that it is dependent on (the interacting CPCs). Adjusted values are computed based on the number of the interacting CPCs that improve (Σ Improved) and reduce (Σ Reduced) human performance. These counts map to regions on the presented graphs that indicate whether an adjusted CPC is improved, reduced, or remain unchanged.

developed by Li and Mosleh (2019), are simulation-based. This means that they can miss system conditions in their analyses and will only ever produce error rate estimates. Thus, we address this shortcoming by integrating CREAM with probabilistic model checking.

It is worth noting that Hollnagel, the inventor of CREAM, has more recently developed a method called the Functional Resonance Analysis Method (FRAM) to represent the complex interactions, dynamics, and failures of sociotechnical systems (Hollnagel, 2017). While Hollnagel himself considers FRAM to be the proper follow-up to CREAM (Hollnagel), FRAM is a qualitative method that does not support the ability to quantify the likelihood of erroneous human behavior. Thus, it cannot properly be considered a HRA.

2.3. Formal methods and probabilistic model checking

Probabilistic model checking comes from the computer science field of formal methods. Formal methods are mathematical languages and techniques for the specification, modeling, and verification of systems (Wing, 1990). Specifications are formulated to rigorously describe desirable system properties, systems are modeled using mathematical languages, and verification mathematically proves whether the model satisfies the specification. Model checking (Clarke et al., 1999), is an automated approach to formal verification, where specification properties (usually represented in a temporal logic) are checked against a state-machine-based model of the system using efficient, exhaustive search algorithms.

A fair amount of research has gone into investigating how formal methods (and especially model checking) can be used to evaluate erroneous human behavior in complex systems (Bolton et al., 2013; Weyers et al., 2017). The vast majority of these analyses are concerned with finding specific unsafe system conditions. However, these methods use non-probabilistic models and are thus not suitable for assessing overall human reliability. Probabilistic model checking offers automated verification techniques for analyzing stochastic systems using probabilistic models (e.g., variants of Markov chains) and probabilistic temporal logic (Kwiatkowska et al., 2007). This enables analysts to both account for probabilistic behavior in their models and prove properties about the probabilities of system behaviors. PRISM (Kwiatkowska et al., 2011) is currently the world's leading open-source software tool for probabilistic model checking. It allows analysts to definitively determine how likely different system behaviors and outcomes are, while accounting for all the different possible behaviors and system dynamics included in a system model. Probabilistic model checking has been used successfully in a number of applications, including systems that rely on human behavior (Feng et al., 2016b,a). Outside of our preliminary work (Zheng et al., 2017), probabilistic model checking has never been used with HRA to account for human error. Despite this, probabilistic model checking is well suited for use with HRA. The discrete, logical nature of probabilistic model checking models is synergistic with the way that HRAs like CREAM compute human error rates. Further, probabilistic model checking can account for system dynamics and interactions between human errors and other types of system errors in a way not previously possible in HRAs.

2.4. Our Previous Work

In preliminary work (Zheng et al., 2017), we demonstrated that probabilistic model checking could be used synergistically with CREAM to predict pharmacy error rates. In particular, this research modeled a simplified version of a pharmacy dispensing procedure along with dynamism associated with pharmacist load based on time of day. We then calculated an average error rate that, when synthesized with expected error rates for unmodeled components, predicted that 1.63% of prescriptions would reach patients with an error. This value was extremely close to the comprehensive 1.7% rate reported by Flynn et al. (2003). However successful this analysis was, it had serious limitations. In particular, it was not able to account for non-human sources of errors or problems (such as prescriptions arriving at the pharmacy with problems), human decisions (and their associated procedure branching

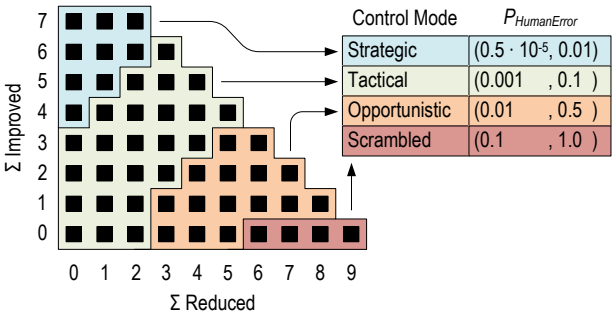

Figure 2: CREAM's method for converting CPC values into control modes (Hollnagel, 1998a). In this, the number of CPCs that improve (Σ Improved) and reduce (Σ Reduced) human performance are both counted. These values map (via the graph) to particular control modes that, in turn, map to ranges of human error probabilities ($P_{HumanError}$). Note that Σ Improved can only go up to 7 because there are two CPCs that can never improve human performance (Goals and Time of Day).

points; such as a pharmacist determining if a prescription is valid and, if not, how to respond in triage), or feedback from patients who notice problems with prescriptions. Additionally, only a limited segment of a pharmacy dispensing procedure was considered. The work presented in this paper addresses all of these shortcomings.

3. Systems Analysis for Formal Pharmaceutical Human Reliability (SAFPHR℞)

In this paper, we extend our previous research (Zheng et al., 2017) to develop the new SAFPHR℞ HRA. SAFPHR℞ combines basic CREAM with probabilistic model checking using PRISM. In doing this, SAFPHR℞ is able to address the major shortcomings of HRA. Specifically, by using probabilistic model checking, our approach can account for interaction between errors and dynamic system behaviors while considering all the possible paths through a modeled system. SAFPHR℞ is able to address the limitations of our previous work (Zheng et al., 2017) by allowing an analyst to model a full pharmacy procedure for filling a prescription as a sequence of tasks. In doing this, SAFPHR℞ accounts for the dynamics of the prescription dispensing process, including the different probabilities that prescriptions arrive at the pharmacy incorrectly, the decisions that pharmacists make, automated equipment used in the process, the sociotechnical factors that influence performance, and the human error rates predicted using these factors. This allows analysts to prove properties about the reliability of procedures.

The application of SAFPHR℞ proceeds as follows. First, the analyst models the procedure he or she wants to evaluate using a well-defined formal modeling architecture. This requires them to describe the procedure and identify which of the CPCs are associated with system dynamism (dynamic CPCs) and which are not (static CPCs). Second, the analyst assesses the CPCs for the different elements of the procedure model and associated dynamic system conditions. He or she then systematically incorporates these into the formal model that is built around the formal architecture. Afterwards, the analyst creates specification properties that allow them to assess the error rates of the overall procedure as well as specific parts of the procedure. The analyst will use probabilistic model checking to evaluate the model in accordance with the asserted specification properties. Finally, The analyst examines the results and explores interventions for addressing discovered problems. This can include modifying the formal system model and/or specifications and running additional model checking analyses to explore the effectiveness of the interventions.

Below we describe how this procedure was realized and illustrate the concepts through the analysis and modeling of a full community pharmacy dispensing procedure.

3.1. Modeling

When modeling for SAFPHR℞ an analyst: represents the pharmacy process as a flowchart; determines which CPCs (Table 1) are static (associated with process tasks) or dynamic (variable based on other environmental criteria) and assesses these; and uses a systematic process to convert this representation into the PRISM language. Below we describe each of these steps.

3.1.1. Flowchart Modeling

Flowchart modeling in SAFPHR℞ uses six basic elements (see Fig. 3): a green circle represents the start of the procedure, a red circle indicates the procedure's end, blue rectangles represent tasks that humans perform in pursuance of the procedure's goals, purple diamonds represent human decision tasks, orange rectangles represent tasks that are performed by automated elements of the system, and arrows represent flow (next steps) from one element to another. Labeled arrows are used as outputs to decision tasks, where the label indicates the next step based on the answer to the question in the decision task.

In this research, the individual tasks included in the workflow for dispensing and delivering drugs to customers (Fig. 3) were identified by the project's subject matter expert Dr. Daly, a practicing pharmacist and Clinical Assistant Professor in the University at Buffalo's School of Pharmacy and Pharmaceutical Sciences. In the procedure modeled for this work, everything starts with a prescription arriving at the pharmacy (via "℞ Arrives at Pharmacy"; which may or may not already contain errors). We grouped elements into four main sub-procedures (shown by light blue shapes with dotted lines; Fig. 3).

The "Verify Prior to Dispensing" procedure allows for checking the validity (e.g. expiration, refills, past use) and legality (e.g. all components for legal prescription are provided and the prescription is not forged) of a given prescription; contacting the prescriber when validity and legality issues are detected; documenting changes for resolving such issues; checking the appropriateness and safety of the prescription for the associated patient (based on things

like drug interactions, drug strength, and directions); contacting the patient or reviewing his or her history when issues are detected; contacting the prescriber when appropriateness and safety issues are detected; and documenting changes for addressing appropriateness and safety issues.

The "Dispense a Prescription" procedure involves filling the prescription. It has the following ordered steps: data entry (inputting information into the computer dispensing database), print label (printing the prescriptions label), get bottle (getting the appropriate stock bottle for the prescribed drug), check the NDC (National Drug Code), count medication (manually measuring or counting out the appropriate amount of the drug by pharmacists, not by machines), attach label (attaching the prescriptions label to the vial), and attach auxiliary label (attaching the auxiliary label to the vial).

In the procedure for "Verify the Final Prescription," the pharmacist must check the validity and legality of the final prescription again and respond appropriately. The pharmacist must also check the appropriateness and safety of the final prescription again for the given patient and respond appropriately. The pharmacist will also check the filled prescription for errors. This involves: checking that the vial has the appropriate labels, checking that the filled prescription's label matches the original prescription, checking that the drug filled is correct, checking that the quantity of the drug filled is correct, and checking that the vial has the appropriate auxiliary labels.

In the last procedure "Deliver Rx to Patient," it includes giving the filled prescription to the correct patient and providing him or her with appropriate counseling.

3.1.2. Assessment of Static and Dynamic CPCs

CPC values associated with each element of a completed procedure model are used by SAFPHRx to compute probabilities of error using techniques from Basic CREAM. Thus, an analyst must assess CPCs for each element of a completed procedure model. In virtue of using probabilistic model checking, SAFPHRx is able to account for both dynamic and static factors that influence human error. SAFPHRx accomplishes this by partitioning CREAM's CPCs (Table 1) into static and dynamic categories. Dynamic CPCs represent factors that will dynamically change in response to dynamic elements of the work environment. Static CPCs have different values associated with each task in the larger procedure (though they can be different between tasks).

Analysts can ultimately decide which CPCs are static and which are dynamic. However, CPCs that are dynamic require that the model represent the dynamic elements of the environment that impact the CPC values. In our current model, Goals, Available Time, and Time of Day are the dynamic CPCs because they will vary based on when a prescription is being dispensed and the number of other tasks happening at the time. As such, their values are impacted by the time of day that a prescription arrives at a pharmacy. We are using the temporal distribution of prescriptions shown in Fig. 4, which is based on real data from a typical Western New York pharmacy. We are currently treating all the following CPCs as static: Organization, Conditions, Support, Procedures, Experience, and Collaboration.

Irrespective of whether a CPC is static or dynamic, its value (reduced, improved, or not significant) must be determined through assessment via a subject matter expert. For static CPCs, this means assessing the values of each static CPC for each task in the procedure. For dynamic CPCs, this means assessing the values of each for each of the possible levels of the dynamic elements. These assessments should be conducted using the standard established by CREAM (Hollnagel, 1998a). For our method, we adapted the CREAM survey to account for the methodological differences introduced by SAFPHRx while using specific language familiar to community pharmacists. Specifically, the procedure in Fig. 3 is used as the context in the survey when assessing static CPCs for each task and the distribution of prescriptions with percentages shown in Fig. 4 served as a general reference for assessing dynamic CPCs.

In the presented work, CPC values were assessed by the project's subject matter expert Dr. Christopher Daly.[1]

3.1.3. The Formal Modeling Architecture and PRISM Modeling

To enable analyst to translate a dispensing procedure model and CPC assessments into a formal model, we developed a formal modeling architecture (see Fig. 5 for an overview). In this, the procedure sub-model represents what step or task is being performed at a given modeled time as well as the logic dictating the order in which procedure tasks are performed. This procedure is modeled as if it is executed using the ordered steps from a procedure model (e.g. Fig. 3). The environmental dynamism sub-model encapsulates dynamic factors that can influence CPCs that are not directly related to the human operator's task (like the time of day from Fig. 4). In our model, this encapsulates the time of day at a community pharmacy, which can influence sociotechnical factors related to time and human load (see Fig. 4). When the procedure is performing a human task, formulas functionally map the state of the procedure and environmental dynamism sub-models to values of the static and dynamic CPCs, respectively. A formula then adjusts the CPC values as specified by CREAM (see Fig. 1) and another formula then uses Basic CREAM (Fig. 2) to compute a probability of error. In the situation where the procedure is performing a non-human task (for example, "Rx Arrives at Pharmacy" from Fig. 3), a formula functionally maps the step to a probability of error which is passed through the "Compute Probability of Error" formula. Finally, procedure compliance uses the

[1] A full listing of the survey and its results can be found at `http://fhsl.eng.buffalo.edu/SAFPHR/`

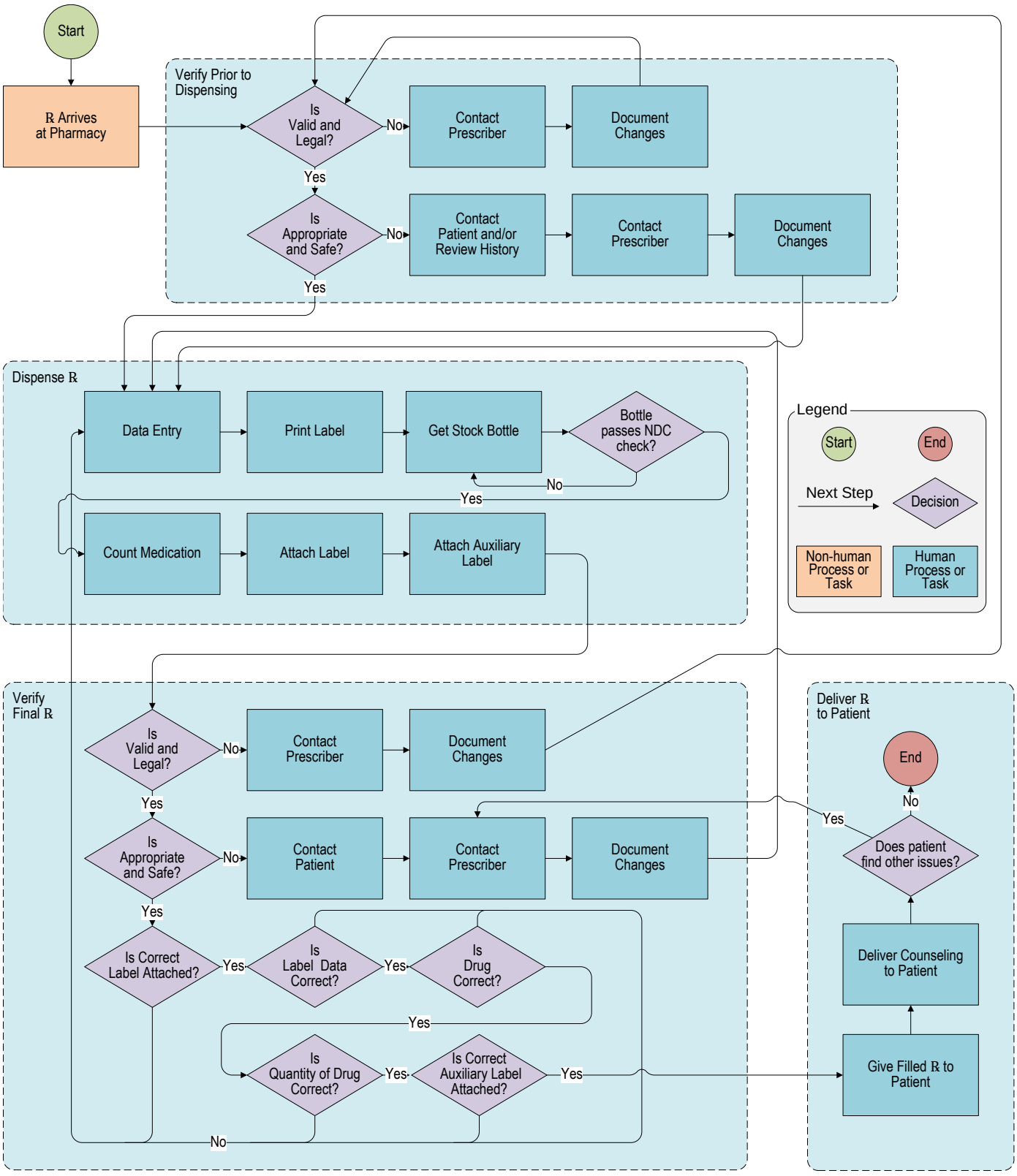

Figure 3: Flow diagram of the community pharmacy dispensing procedures analyzed with SAFPHR℞. The start and end of the procedure are represented as circles. Human decision are diamonds. Human tasks are rectangles. Arrows show the next step in the procedure. Arrows out of decisions indicate the next step based on the answer to the question in the decision node.

computed probability of error to determine whether the given step in the modeled procedure is performed "correct" or "incorrect." The procedure can examine the state of the procedure compliance to influence subsequent procedure performance. For example, a pharmacist can check whether an element of a prescription was filled correctly when performing a decision task. The state of procedure compliance could also be used to influence modeled environmental dynamism, but this is currently not used in our instantiation of the architecture.

This architecture (Fig. 5) can be used to implement a dispensing procedure model using PRISM's input language (Parker et al., 2017). For this project, we instantiated this architecture for the procedure and dynamics from Fig. 3 and Fig. 4 and their associated CPC assessments as a discrete-time Markov chain (DTMC). Figures 6 and 7 provide an illustration of how this model was implemented.[2]

The PRISM code starts (Fig. 6) by defining constants that are used in later model concepts. `Improved`, `NotSignificant`, and `Reduced` in `line 4` are integer constants representing the three levels of CPCs. `Strategic`, `Tactical`, `Opportunistic`, and `Scrambled` from `line 6` to `line 7` represent the error probabilities associated with CREAM's control modes (Fig. 2). In `line 9`, `Incorrect`, `NotApplicable`, and `Correct` respectively define constants for representing whether a task was performed incorrectly, not performed yet (or at all), or correctly. The `Correct` and `Incorrect` constants are also used by the model to represent whether parts of a filled prescription contains any errors. In `line 12`, `Start—End` define a unique numerical ID for each of the elements from the procedure model (Fig. 3). In special circumstances, such as the non-human task "℞ Arrives at Pharmacy" from Fig. 3, the element is given two IDs (in this case `RxArrives1` and `RxArrives2`) because it needs to account for two different factors in the models: the validity and legality of the arriving prescription as well as its appropriateness and safety. On `line 15`, constants define probabilities for prescriptions arriving with validity and legality issues (`P_V`), arriving with appropriateness and safety problems (`P_S`), and the probability that a patient will discover issues with a delivered prescription that contains errors (`P_PatientFindsIssues`). Note that `p1` and `p2` are placeholders for actual probabilities that are inserted by the analyst and will be discussed in more depth subsequently. The probability that a patient discovers issues with a prescription (`P_PatientFindsIssues = 12/33`) comes from Witte and Dundes (2007).

Constant definition is followed by the definition of two synchronously composed modules (Fig. 6). These represent the discrete-time Markov chain behavior of the model for both the environmental dynamism of the model (`TimePeriod`) and the performance of the dispensing procedure (`Procedure`).

The `TimePeriod` module is shown in `lines 18—22` of Fig. 6. In this, the `t` variable is used to represent which of the eight possible time periods that a prescription arrives in from Fig. 4. The transition assignment (`lines 20` and `21`) is designed so that, in the first state of the larger model, the time period is assigned based on the distribution of prescription arrivals. For example, when `t = 1`, the prescription filled rate is 12% from 6 AM to 8 AM and when `t` = 2, the prescription filled rate is 18% from 8 AM to 10 AM. As such, there is a 12% of chance that the values of `t` will be 1, 18% of chance that the values of `t` will be 2. The net effect of this is that analyses will account for the effect of all the different time periods and the probabilities of prescriptions being filled in those times.

The procedure module is presented in `lines 25—74` of Fig. 6. This has a single variable `ProcedureStep` (`line 26`) to represent the element of procedure being performed. The variables that follow (`lines 28—40`) represent each task from the procedure indicating if the associated element has been performed correctly (`Correct`), incorrectly (`Incorrect`), or not performed (`NotApplicable`; the default value). This includes `RxValidLegal` and `RxAppSafe` which account for non-human source of errors and (for the purposes of our implementation) whether the prescription arrives with errors. All the other `Task_` variables are generic ones that we use here to show how different SAFPHR℞ features are implemented.

Variable definitions are followed by transition logic. The first transition (`line 42` and `43`) is designed to start the performance of the procedure in that it requires that `t` (from the `TimePeriod` module) be assigned (`t > 0`). If this is true, then, with probability 1, the procedure step will be set to the ID of the first task, in this case `RxArrives1`. The two transitions that follow on `lines 45—50` illustrates how a non-human task is represented. Note that because the task in question ("℞ Arrives at Pharmacy"; Fig. 3) can determine whether a prescription arrives with errors in two different ways, its behavior is spread over two transi-

[2] A full listing of model code can be found at `http://fhsl.eng.buffalo.edu/SAFPHR/`.

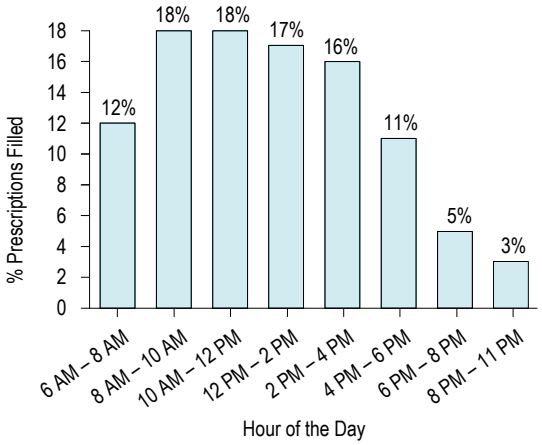

Figure 4: Graph showing how the distribution of prescriptions filled at a WNY pharmacy. There are eight zones; each zone represents a two-hour timespan besides the final zone, which represents a three-hour timespan. Each time zone has a corresponding bar on the graph, each indicates the average percentage of prescriptions filled during that time zone.

tions. The first determines if the prescription has validity and legality issues. In this, with a probability of P_V (a constant from above), RxValidLegal will be Incorrect: the prescription will be invalid or illegal. Similarly, with a probability of 1 - P_V, the prescription will arrive without errors and RxValidLegal will be Correct. In both situations, the ProcedureStep will move to the second part of this task. The second transition for the task uses the same logic to determine if the prescription is appropriate and safe.

The transitions from lines 54–64 in Fig. 6 illustrate different types of generic task behavior. Because all of these represent human tasks, the probability of the task being performed with an error (ProbError) is computed using a formula shown in Fig. 7 and discussed subsequently. An example that illustrates how two transitions are used to represent a decision task are given in lines 54–59. In these, DecisionCriterion is a placeholder for a Boolean expression that will evaluate to true when a given decision task should produce a "Yes" outcome. The two transitions allow for the two possible conditions that could arise in the execution of the model where DecisionCriterion is true and the person should decide on "Yes" or it is false (!DecisionCriterion) and he or she should decide "No," with errors potentially made under both conditions. For example, when the prescription has a validity issue like its refill date being incorrect, the DecisionCriterion for the step of "Is valid and Legal?" will be "The prescription contains validity or legality issues." Note that across these two transitions, whether the DecisionCriterion is true and whether the person makes an error will determine which procedure is performed next. In contrast to a decision task, a transition for a standard human task (lines 62–64) only require one transition, does not have a DecisionCriterion, and only ever proceeds to one next step.

The last set of transitions in our module (lines 69–73) represent the transitions that can complete the prescription filling process by moving the ProcedureStep to End. These transitions are also unique in that they account for the situation where a patient may discover an error in the filled prescription and give it back to the pharmacist or accept it (with or without errors). In the transitions, ErrorInFilledPrescription is a Boolean expression indicating errors exists in the filled prescription. It will be true if the prescription arrived with validity, legality, appropriateness, or safety issues (that were not caught and corrected) or if any of the human tasks from the "Dispense Rx" sub-model were performed incorrectly. Furthermore, if the prescription contains any error, there is a set probability that the patient will identify it (P_PatientFindsIssues in Fig. 6). Thus, under ProcedureStep = g and ErrorInFilledPrescription, with a probability of P_PatientFindsIssues(a constant discussed above), the patient discovers an error from the filled prescription thus the ProcedureStep will move back to a middle step. In all other cases, the filled prescription will be delivered to the patient.

Figure 7 lists all the formulas that are used to calculate the probabilities of human error (ProbError) used in the transitions from Figure 6. From lines 77–80, the three formulas compute the three dynamic CPC values based on the time period t using values assessed from the subject matter expert. For example, in the TimeOfDay formula, if t is equal to 5, 7, or 8, the value of "TimeOfDay" will be Reduced; otherwise, the value of "TimeOfDay" will be NotSignificant. The formulas in lines 84–line 89 determine the values of the static CPCs at the current procedure step based on values assessed from the subject matter experts. For example on line 84, if ProcedureStep = i, the value of Organization at step i will be Organization_i. Note that i represents a placeholder for any given procedure step and Organization_i is the assessed value of the organization CPC for that step. Lines 93–line 96, shows an example of how the CPC adjustments from Fig. 1 are performed (in this case the Collaboration CPC). In CollaborationAdj, if both the Organization and Experience formulas evaluate to Improved, the value of CollaborationAdj will be Improved. If both are Reduced, CollaborationAdj will be Reduced; otherwise,

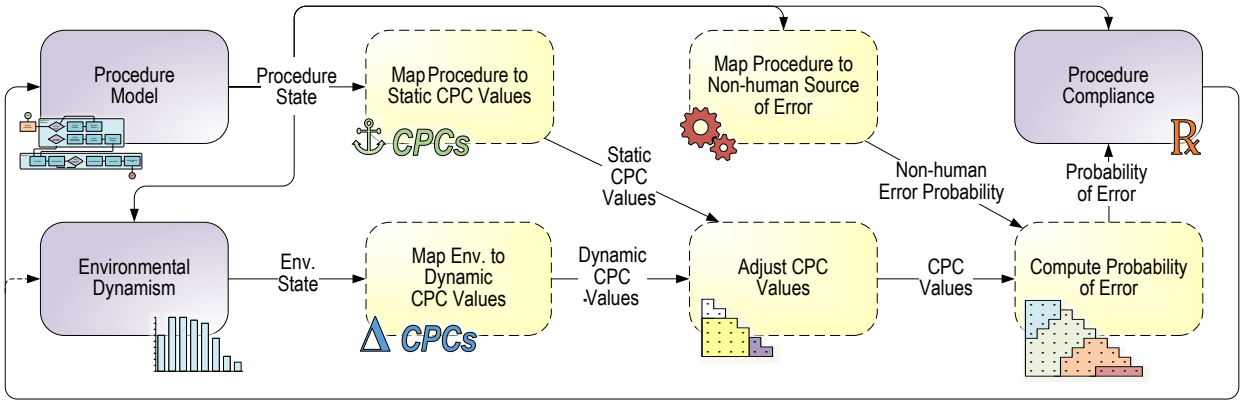

Figure 5: Overview of the formal modeling architecture used in SAFPHRx. Shapes with solid lines are formal sub-models. Shapes with dotted lines are formulas (functions that compute values using state variables from sub-models and other formulas). Arrows describe variables shared between sub-models and formulas. Dotted arrows represent shared variables that are possible with the architecture, but currently not used in our model.

```
dtmc // discrete-time Markov chain

// Constant definitions
const int Improved = 1; const int NotSignificant = 0; const int Reduced = -1; // CPC level constants

const double Strategic     = 0.000005; const double Tactical  = 0.001;        // Control mode probabilities
const double Opportunistic = 0.01;     const double Scrambled = 0.1;          // Values are lower bounds

const int Incorrect = -1; const int NotApplicable = 0; const int Correct = 1;  // Task performance status

// Each procedure node (start, end, task, or decision) has a unique id with End being the total number of nodes
const int Start = 1; const int RxArrives1 = 2; const int RxArrives2 = 3; ... const int End   = 32;

// Constant probabilities used by non-human sources of error and decisions not made by the pharmacist
const double P_V = p1; const double P_S = p2; const double P_PatientFindsIssues = 12/33;

// Environmental dynamism, eight time zones
module TimePeriod
    t:[0..8] init 0;
    [] t = 0 -> 0.12 : (t' = 1) + 0.18 : (t' = 2) + 0.18 : (t' = 3) + 0.17 : (t' = 4) + 0.16 : (t' = 5)
                + 0.11 : (t' = 6) + 0.05 : (t' = 7) + 0.03 : (t' = 8);
endmodule

// Procedure sub-model
module Procedure
    ProcedureStep : [Start..End]         init Start;         // Variable representing the current step in the procedure
    ...
    RxValidLegal  : [Incorrect..Correct] init NotApplicable; // Variable showing if the prescription is valid and legal
    RxAppSafe     : [Incorrect..Correct] init NotApplicable; // Variable showing if the prescription is appropriate and safe
    ...
    Task_a,       : [Incorrect..Correct] init NotApplicable; // There is a unique Task_ variable for every task in the
    Task_b,       : [Incorrect..Correct] init NotApplicable; // modeled procedure. a, b, ..., h represent
    Task_c,       : [Incorrect..Correct] init NotApplicable; // the unique procedure step (with assumed values between
    Task_d,       : [Incorrect..Correct] init NotApplicable; // Start and End) associated with the task
    ...
    Task_e,       : [Incorrect..Correct] init NotApplicable;
    Task_f,       : [Incorrect..Correct] init NotApplicable;
    ...
    Task_g,       : [Incorrect..Correct] init NotApplicable;
    Task_h,       : [Incorrect..Correct] init NotApplicable;
    ...
    [] t > 0 & (ProcedureStep = Start) ->
        1                      :  (ProcedureStep' = RxArrives1);
    // The first transition: in our model this represents a task with a non-human source of error
    [] (ProcedureStep = RxArrives1) ->
        1 - P_V  : (RxValidLegal' = Correct)   & (ProcedureStep' = RxArrives2)
        + P_V    : (RxValidLegal' = Incorrect) & (ProcedureStep' = RxArrives2);
    [] (ProcedureStep = RxArrives2) ->
        1 - P_S     : (RxAppSafe' = Correct)      & (ProcedureStep' = a)
        + P_S       : (RxAppSafe' = Incorrect)    & (ProcedureStep' = a);
    ...
    // Form of the transitions for a decision task:
    // DecisionCriterion is a Boolean expression indicating if the correct decision is "yes"
    [] (ProcedureStep = b) & DecisionCriterion ->
        1 - ProbError          : (Task_b' = Correct)       & (ProcedureStep' = c)
        + ProbError            : (Task_b' = Incorrect)     & (ProcedureStep' = d);
    [] (ProcedureStep = b) & !DecisionCriterion ->
        1 - ProbError          : (Task_b' = Correct)       & (ProcedureStep' = d)
        + ProbError            : (Task_b' = Incorrect)     & (ProcedureStep' = c);
    ...
    // Form of the transitions for a human task:
    [] (ProcedureStep = e) ->
        1 - ProbError          : (Task_e' = Correct)       & (ProcedureStep' = f)
        + ProbError            : (Task_e' = Incorrect)     & (ProcedureStep' = f);
    ...
    // The transitions that account for patients discovering an error or accepting the filled prescription
    // No additional transitions can occur once ProcessStep = End because the prescription has been successfully delivered
    // to the patient
    [] (ProcedureStep = g) & ErrorInFilledPrescription ->
        1 - P_PatientFindsIssues : (ProcedureStep' = End)
        + P_PatientFindsIssues   : (ProcedureStep' = h);
    [] (ProcedureStep = g) & !ErrorInFilledPrescription ->
        1                        : (ProcedureStep' = End);
endmodule
...
```

Figure 6: Example model code (started here and continued in Fig. 7) for implementing the architecture from Fig. 5 with details specifically related to the procedure from Fig. 3. Code is presented using PRISM's modeling language (Parker et al., 2017). Text following // indicate comments and are presented in light blue. PRISM reserved language words are magenta. Named constants (i.e. Improved) are green. Modules (like TimePeriod) are dark yellow. Variables (i.e. ProcedureStep) are blue. Formulas like ProbError (quantities that are dynamically computed in each state from constants, variables, and other formulas) are orange. Items in red represent placeholders for code or values that would be inserted to represent generic conditions that the analyst would manually specify in a complete model. For example b represents the step number of a given task, presented above with a generic name of Task_b and DecisionCriterion represents a Boolean expression indicating when a given decision task should evaluate to "yes."

```
75  ...
76  // Computing the Dynamic CPC Values Based on the Time Period (t)
77  formula TimeOfDay         = (t = 5 | t = 7 | t = 8) ? Reduced : NotSignificant;
78  formula SimultaneousGoals = (t = 1 | t = 2)         ? Reduced : NotSignificant;
79  formula AvailableTime     = (t = 1)                 ? Reduced :
80                              (t = 3 | t = 4 | t = 6) ? NotSignificant : Improved;
81
82  // Static CPC Values at the Current Procedure Step (i). Note that Organization_i, Conditions_i, Support_i,
83  // Procedures_i, Experience_i, and Collaboration_i are Specific Values Determined by CPC Assessment
84  formula Organization   = ... ProcedureStep = i ? Organization_i  : ... ;
85  formula Conditions     = ... ProcedureStep = i ? Conditions_i    : ... ;
86  formula Support        = ... ProcedureStep = i ? Support_i       : ... ;
87  formula Procedures     = ... ProcedureStep = i ? Procedures_i    : ... ;
88  formula Experience     = ... ProcedureStep = i ? Experience_i    : ... ;
89  formula Collaboration  = ... ProcedureStep = i ? Collaboration_i : ... ;
90
91  // CPC Adjustment. Here we only show the adjustment for Collaboration. Other adjustments are similar.
92  ...
93  formula CollaborationAdj = ((Organization = Improved ? 1 : 0)
94                              + (Experience = Improved ? 1 : 0)) = 2 ? Improved :
95                             ((Organization = Reduced  ? 1 : 0)
96                              + (Experience = Reduced  ? 1 : 0)) = 2 ? Reduced  : Collaboration;
97  ...
98
99  // Calculating the Probability of Error
100 formula NumRed  = (Organization       = Reduced  ? 1 : 0) + (TimeOfDay        = Reduced  ? 1 : 0)
101                 + (GoalsAdj          = Reduced  ? 1 : 0) + (AvailableTimeAdj = Reduced  ? 1 : 0)
102                 + (ConditionsAdj     = Reduced  ? 1 : 0) + (Support          = Reduced  ? 1 : 0)
103                 + (Procedures        = Reduced  ? 1 : 0) + (Experience       = Reduced  ? 1 : 0)
104                 + (CollaborationAdj = Reduced  ? 1 : 0);
105 formula NumImp  = (Organization       = Improved ? 1 : 0) + (TimeOfDay        = Improved ? 1 : 0)
106                 + (GoalsAdj          = Improved ? 1 : 0) + (AvailableTimeAdj = Improved ? 1 : 0)
107                 + (ConditionsAdj     = Improved ? 1 : 0) + (Support          = Improved ? 1 : 0)
108                 + (Procedures        = Improved ? 1 : 0) + (Experience       = Improved ? 1 : 0)
109                 + (CollaborationAdj = Improved ? 1 : 0);
110 formula ProbError = (NumRed   =  0 & NumImp >= 4) | (NumRed = 1 & NumImp >= 5) | (NumRed = 2 & NumImp >= 6)
111                   ? Strategic :
112                   (NumRed   =  0 & NumImp <  4) | (NumRed = 1 & NumImp <  5) | (NumRed = 2 & NumImp <  6)
113                   | (NumRed =  3 & NumImp >= 2) | (NumRed = 4 & NumImp >= 3) | (NumRed = 5 & NumImp >= 4)
114                   ? Tactical :
115                   (NumRed   =  3 & NumImp <  2) | (NumRed = 4 & NumImp <  3) | (NumRed = 5 & NumImp <  4)
116                   | (NumRed >= 6 & NumImp >= 1)
117                   ? Opportunistic : Scrambled;
```

Figure 7: Example model code (continued from Fig. 6) for implementing the architecture from Fig. 5. See Fig. 6 for a description of the syntax.

no adjustment is made. Additionally, `CollaborationAdj` will be the value of `Collaboration`.

The formulas `NumRed` and `NumImp` (`lines 100–109`) are used to count the number of CPCs (post adjustment) rated as `Reduced` or `Improved`, respectively. The `ProbError` formula (`lines 110–117`) uses these values to compute a probability of error (the previously discussed constants, `Strategic`, `Tactical`, `Opportunistic`, and `Scrambled`) based on the associated control mode in accordance with the algorithm from Fig. 2.

3.1.4. Specification Properties

With a completed formal model, an analyst must check specification properties to compute the probability of different outcomes or prove other properties. SAFPHR currently uses a number of specification patterns to assert properties of interest to analysts. These specifications are formulated using probabilistic temporal logic (Parker et al., 2017). Below we present several of these specification property patterns.[3] Note that all of these properties are designed to compute reliability values in the failure domain: compute the probability of errors or failures.

To assess the overall reliability of a pharmacy's procedure (its overall dispensing error rate) we can check a property in following form:

Procedure Eventual Reliability:

$$P =? \left[\mathbf{F} \left(\begin{array}{l} (\mathit{ProcedureStep} = \mathit{End}) \\ \wedge \left(\begin{array}{l} (\mathit{RxValidLegal} = \mathit{Incorrect}) \\ \vee (\mathit{RxAppSafe} = \mathit{Incorrect}) \\ \vee_{t \in T} (t = \mathit{Incorrect}) \end{array} \right) \end{array} \right) \right]. \quad (1)$$

This tells PRISM to calculate the probability ($P =?$) that the prescription eventually ($\mathbf{F}$) has an uncorrected error that it arrived with ($(\mathit{RxValidLegal} = \mathit{Incorrect}) \vee (\mathit{RxAppSafe} = \mathit{Incorrect})$) or any of the human tasks from the "Dispense Rx" and "Deliver Rx to Patient" sub-models ($t \in T$) performed incorrectly when the prescription is delivered to the patient ($\mathit{ProcedureStep} = \mathit{End}$). Note that in this and other property patterns, the word "Eventual" is used in the pattern's title to indicate that reliability is computed based on when the prescription has reached a patient.

We can also calculate error rates of each of a procedure's

[3] A full listing of specification properties checked in this research can be found at `http://fhsl.eng.buffalo.edu/SAFPHR/`.

sub-models (i.e. "verify prior to dispensing," "dispense a prescription," "verify the final prescription," and "deliver drug to patient") using a specification of the form:

$$\text{Submodel Eventual Reliability:}\quad P=?\left[\mathbf{F}\left(\begin{array}{l}(\mathit{ProcedureStep}=\mathit{End})\\ \wedge\left(\bigvee_{s\in S}(s=\mathit{Incorrect})\right)\end{array}\right)\right]. \tag{2}$$

In this, PRISM will only calculate the probability that eventually any of the task in a given sub-model ($s \in S$) were performed incorrectly once the prescription is delivered to the patient.

We have also identified a property pattern for calculating the probability that eventually a given task (g) will have been performed incorrectly once the prescription has been delivered to the patient:

$$\text{Task Eventual Reliability:}\quad P=?\left[\mathbf{F}\left(\begin{array}{l}(\mathit{ProcedureStep}=\mathit{End})\\ \wedge(g=\mathit{Incorrect})\end{array}\right)\right]. \tag{3}$$

The previous specification property patterns calculate error rates based on whether an error is still apparent once the prescription is delivered to a patient. However, by being coupled with the requirement that $\mathit{ProcedureStep} = \mathit{End}$, they do not give insights into the probability that an error ever occurs because errors could be corrected due to the feedback in the procedure. Thus, we also formulated patterns for checking "General" reliability.

The first of these patterns checks the overall reliability of a given sub-model by determining if any of its contained tasks ($s \in S$) are ever incorrect:

$$\text{Submodel General Reliability:}\quad P=?\left[\mathbf{F}\left(\begin{array}{l}(\mathit{ProcedureStep}=\mathit{NextStep})\\ \wedge\left(\bigvee_{s\in S}(s=\mathit{Incorrect})\right)\end{array}\right)\right]. \tag{4}$$

The second checks the overall reliability of a given task (g) ever being incorrect:

$$\text{Task General Reliability:}\quad P=?\left[\mathbf{F}\left(\begin{array}{l}(\mathit{ProcedureStep}=\mathit{NextStep})\\ \wedge\left(\ g=\mathit{Incorrect}\ \right)\end{array}\right)\right]. \tag{5}$$

Note that both Eqs. (4) and (5) have the requirement that $\mathit{ProcedureStep} = \mathit{NextStep}$. In these, *NextStep* represents the ID of the task that immediately follows the task or sub-model that is the subject of the analysis. For example, if the reliability of the "Data Entry" task is being assessed, *NextStep* would correspond to the ID of "PrintLabel." If the reliability of the "Dispense Rx" sub-model is being checked, *NextStep* would correspond to the ID of the "IsValidandLegal?" task. Doing this ensures that the model checker is evaluating the state of the target element immediately following its execution.

The specifications from Eqs. (2) to (5) all give insights into inefficiencies of the procedure by showing which errors are most likely to occur. However, due to feedback in the procedures and redundant checks in the "Verify Final Rx" submodel, probabilities that will be acquired by checking these will not necessarily indicate how and where intervention should be deployed to improve reliability. To address this, SAFPHRx supports the ability allowing analysts to modify the system model and rerun analyses. For example, an analyst can identify how much a particular task (with a `ProcedureStep` of `i`) is contributing to an error rate by modifying `ProbError` from Fig. 7 so that it is always 0 for that task. Figure 8 illustrates how this is accomplished. The analyst can then recheck the original sections and compare how the change impacted the resulting error rates.

4. Methods

Using our formulation of SAFPHRx with the CPC assessments from our subject matter expert, we evaluated the reliability of a typical Unites States community pharmacy (based on data from Western New York pharmacies) that does not use automated dispensing equipment. Because SAFPHRx uses basic CREAM, human reliability predictions can have a range of values (Fig. 2). To accommodate this, all the SAFPHRx analyses actually make use of two different versions of the model. One uses the minimum probability from each control mode and the other uses the maximum. In either case, the values used in the model are assigned to the `Strategic`, `Tactical`, `Opportunistic`, and `Scrambled` constants from the model. For example, the minimum probabilities are used in the code shown in Fig. 6.

To calculate reliability values, specifications asserted using the property patterns in Eqs. (1) to (5) were checked using the PRISM model checker on a desktop computer with a 3.70 GHz Xeon processor and 128 GB of RAM running Linux Mint. Equation (1) (Procedure Eventual Reliability) was used to check the overall error rate of the procedure. Multiple instances of Submodel Eventual Reliability (Eq. (2)) and Task Eventual Reliability (Eq. (3)) were used to determine the probability that each submodel and task (respectively) was done incorrectly once a prescription reached a patient. Instances of the respective General Reliability specifications from Eqs. (4) and (5) were also checked. Finally, we used the technique described at the end of Section 3.1.4 to assess the impact of eliminating the erroneous behavior associated with any given task on overall reliability. This was done by preventing each task from being performed erroneously in a given model, checking Procedure Eventual Reliability (Eq. (1)), and subtracting this from the Procedure Eventual Reliability observed for the unmodified model.

In all of these analyses, the predicted values from the minimum and maximum models were combined into point estimates using the log average method of probability estimation (Clemens and Simmons, 1998):

$$P_{PointEstimatation} = 10^{(log(P_{Lower})+log(P_{Upper}))/2}. \tag{6}$$

Finally, two observational studies by Gilligan et al. (2012) and by Odukoya et al. (2015) suggested that 11%

```
110 formula ProbError = (ProcedureStep = i )? 0 : ...
```

Figure 8: Modification to ProbError to allow the given task with a ProcedureStep of i to not exhibit errors. Note that in this ... represents the original formulation of ProbError from Fig. 7.

of prescriptions have at least one problem that requires intervention by pharmacist. However, we could not find statistics to differentiate between issues of validity and legality and problems with appropriateness and safety. Thus, we conducted a sensitivity analysis when computing the overall procedure's error rate (using the specification for Procedure Eventual Reliability; Eq. (1)) to study the impact of p1 and p2 (see Fig. 6) on the results. This was done by letting p1 and p2 assume any possible combination of values from 0.1 to 0.9 in increments of 0.1. Results of these sensitivity analyses were then used to influence the other analyses conducted with our models.

5. Results

Results of our analyses of Procedure Eventual Reliability (Eq. (1)) with the variable values of P_V and P_S are shown in Table 2, where we report both the upper (U) and lower (L) limits along with the resulting log average (A). These results showed that the values of P_V and P_S had very little impact on the error rate, with a maximum difference between the computed point estimates (the log average) of 0.0000367. Because these parameters had very little effect, we performed all the subsequent analyses using fixed values of P_V = 0.1 and P_S = 0.1. Thus, with P_V = 0.1, P_S = 0.1, we get an average Procedure Eventual Reliability error rate of 0.0005436.

Table 3 reports the log average results for both Submodel Eventual Reliability and Submodel General Reliability. These showed that while "Verify Prior to Dispensing" had the highest error rate in its Submodel Eventual Reliability (0.0115984), the "Dispensing a prescription" portion of the procedure had the worst Submodel General Reliability (0.0255339).

The log average results for Task Eventual Reliability and Task General Reliability for each task are shown in Table 4. Table 4 also contains results under "New Prob," which represents the Procedure Eventual Reliability associated with the task always being performed correctly. This is reported with "Improvement," which indicates how much of a reliability improvement (over the original Procedure Eventual Probability of 0.0005436) "New Prob" produced. Both of these were computed using the method discussed in the last paragraph of Section 3.1.4. Across these results, bold entries are used to highlight values with values orders of magnitude higher than others from the same column.

6. Discussion

This paper introduced the new, formal HRA SAFPHRx for use in the analysis of community pharmacy procedures. By grounding the HRA in CREAM (Hollnagel, 1998a) and the PRISM probabilistic model checker (Kwiatkowska et al., 2011), SAFPHRx allows us to account for dynamic system conditions and compute accurate error rates at a level that was not previously possible in HRA. The presented application demonstrates the feasibility and power of SAFPHRx in predicting the medication error rates accurately. The error rate of 0.05436% predicted by SAFPHRx is extremely close to the 0.057% rate reported by Szeinbach et al. (2007). This rate was one of the lowest reported in the literature, but was from one of the most recent studies on the subject which helps account for the electronic prescribing technology that is now employed in most pharmacies. Thus, this provides good evidence that our analyses are valid.

Using the same math that IOM (2006) applied to the 1.7% error rate we discussed in the introduction; our 0.05436% value corresponds to approximately 1,630,800 pharmacy errors a year in the United States, 116,486 which are clinically significant. This, in turn, translates into 46 errors per pharmacy per year, 3 of which are clinically significant. Thus, while 0.05436% is substantially smaller than 1.7%, it still constitutes a major risk to human health and safety.

The power of our approach is further found in the results we obtained from our analyses of eventual and general reliability of each submodel and each task in the procedure. This gives us information about the efficiency of the procedure. In the analyses of the submodels, the highest error rates were observed for Submodel General Reliability for the "Verify Prior to Dispensing" and "Dispense Rx" submodels (see Table 3). This suggests that analysts wishing to improve the efficiency of their dispensing procedure could focus on improving the reliability of these two processes. Note that the error rates of Submodel Eventual Reliability for all submodels (except "Delivering Rx to Patient", for which Eventual and General reliability are the same concept) decreased from their respective Submodel General Reliability rates. This indicates that the procedure is able to correct errors after they are performed. The largest difference was seen for "Dispense Rx", which saw an order of magnitude improvement from General to Eventual. This is likely due to the fact that errors caused under this submodel have the potential to be detected under "Verify Final Rx" and sent back to "Dispense Rx" for correction (see Fig. 3). Feedback related to problems originating in the other models can still occur to correct prescription errors, but this may not necessarily result in the reperformance of the original, incorrectly performed task. For example, assume an error occurs for "Contact Prescriber" (for addressing appropriateness and safety issues) in the "Verify Prior to Dispensing" submodel, and this error is caught

Table 2: Procedure Eventual Reliability error rates of the pharmacy procedure (Fig. 3) with variable initial probabilities of validity and legality issues (P_V) as well as probabilities of appropriateness and safety (P_S)

		P_V								
P_S		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.1	U	0.0288563	0.0291925	0.0295288	0.0298650	0.0302013	0.0305375	0.0308738	0.0312100	0.0315463
	A	0.0005436	0.0005482	0.0005527	0.0005573	0.0005618	0.0005663	0.0005708	0.0005754	0.0005799
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107
0.2	U	0.0288913	0.0292240	0.0295566	0.0298892	0.0302218	0.0305544	0.0308871	0.0312197	0.0315523
	A	0.0005440	0.0005485	0.0005530	0.0005575	0.0005620	0.0005665	0.0005710	0.0005754	0.0005799
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107
0.3	U	0.0289264	0.0292554	0.0295844	0.0299134	0.0302423	0.0305713	0.0309003	0.0312293	0.0315583
	A	0.0005443	0.0005488	0.0005533	0.0005577	0.0005622	0.0005666	0.0005711	0.0005755	0.0005800
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107
0.4	U	0.0289615	0.0292868	0.0296122	0.0299375	0.0302629	0.0305882	0.0309136	0.0312389	0.0315643
	A	0.0005447	0.0005491	0.0005535	0.0005580	0.0005624	0.0005668	0.0005712	0.0005756	0.0005800
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107
0.5	U	0.0289965	0.0293182	0.0296400	0.0299617	0.0302834	0.0306051	0.0309268	0.0312485	0.0315703
	A	0.0005450	0.0005494	0.0005538	0.0005582	0.0005626	0.0005670	0.0005714	0.0005757	0.0005801
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107
0.6	U	0.0290316	0.0293497	0.0296677	0.0299858	0.0303039	0.0306220	0.0309401	0.0312582	0.0315763
	A	0.0005453	0.0005497	0.0005541	0.0005584	0.0005628	0.0005671	0.0005715	0.0005758	0.0005802
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107
0.7	U	0.0290666	0.0293811	0.0296955	0.0300100	0.0303244	0.0306389	0.0309533	0.0312678	0.0315822
	A	0.0005457	0.0005500	0.0005543	0.0005587	0.0005630	0.0005673	0.0005716	0.0005759	0.0005802
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107
0.8	U	0.0291017	0.0294125	0.0297233	0.0300341	0.0303450	0.0306558	0.0309666	0.0312774	0.0315882
	A	0.0005460	0.0005503	0.0005546	0.0005589	0.0005632	0.0005675	0.0005717	0.0005760	0.0005803
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107
0.9	U	0.0291367	0.0294439	0.0297511	0.0300583	0.0303655	0.0306727	0.0309799	0.0312870	0.0315942
	A	0.0005463	0.0005506	0.0005549	0.0005591	0.0005634	0.0005676	0.0005719	0.0005761	0.0005804
	L	0.0000102	0.0000103	0.0000103	0.0000104	0.0000105	0.0000105	0.0000106	0.0000106	0.0000107

In the above, U, A, and L represent the upper bound, log average, and lower bound respectively.

Table 3: Submodel Eventual Reliability (Eq. (2)) and Submodel General Reliability (Eq. (4)) for each submodel from Fig. 3.

	Submodel Reliability	
Submodel	Eventual	General
Verify Prior to Dispensing	0.0115984	0.0167122
Dispense Rx	0.0002468	0.0255339
Verify Final Rx	0.0001001	0.0001486
Delivering Rx to Patient	0.0002862	0.0002862

during verification in "Verify Final Rx". In this situation, the prescriber will be recontacted and the prescription will ultimately be returned to the "Dispense Rx" submodel and not the "Verify Prior to Dispensing" one. Thus, the original misperformance of "Contact Prescriber" (for addressing appropriateness and safety issues) will never be corrected in the model.

The results for Task Eventual Reliability and Task General Reliability (Table 4) are largely consistent with those from the submodel analyses. Specifically, the highest error rates were observed for tasks from "Verify Prior to Dispensing" under Eventual Reliability and from "Verify Prior to Dispensing" and "Dispense Rx" under General Reliability. These results are helpful because they indicate where process improvements could be made to improve efficiency in each of these submodels. Large differences between respective General and Eventual reliability values are likely due to the same feedback mechanisms discussed for the submodel analyses.

The Eventual and General reliability values for submodels and tasks give analysts insights into the efficiency of pharmacists performing the dispense procedure. However, the complexity of the procedure makes it difficult to determine how to best address reliability issues that ultimately reach patients. Thus, another benefit of SAFPHRx is that it gives analyst the ability to explore the effectiveness of interventions (see Improvement from Table 4). These results showed that "Print Label," "Count Medication," "Is Quan-

Table 4: Task Eventual Reliability (Eq. (3)), Task General Reliability (Eq. (5)), Procedure Eventual Probability without task error contributions (New Prob), and its Improvement over original Procedure Eventual Reliability.

	Task Reliability			
Single Task	Eventual	General	New Prob.	Improvement
"Is Valid and Legal?"	0.0000043	**0.0106685**	0.0005390	0.0000046
"Contact Prescriber"	**0.0024652**	**0.0025878**	0.0005436	< 1E-07
"Document changes" (validity and legality issues)	0.0000003	0.0000003	0.0005436	< 1E-07
"Is Appropriate and Safe?"	**0.0099640**	**0.0097124**	0.0005433	0.0000003
"Contact Patient and/or Review history" (appropriateness and safety issues)	0.0000313	0.0000315	0.0005436	< 1E-07
"Contact Prescriber" (appropriateness and safety issues)	**0.0013988**	**0.0014097**	0.0005436	< 1E-07
"Document Changes"(appropriateness and safety issues)	0.0000299	0.0000315	0.0005436	< 1E-07
"Data Entry"	0.0000015	0.0003013	0.0005369	0.0000067
"Print Label"	0.0001435	0.0003022	0.0003989	**0.0001447**
"Get Stock Bottle"	< 1E-07	**0.0135775**	0.0005430	0.0000006
"Bottle Passes NDC check?"	< 1E-07	0.0001486	0.0005430	0.0000006
"Count Medication"	0.0000709	**0.0125366**	0.0004509	**0.0000927**
"Attach Label"	0.0000015	**0.0124991**	0.0005373	0.0000063
"Attach Auxillary Label"	0.0000002	**0.0021200**	0.0005424	0.0000012
"Is Valid and Legal?" (final check, validity and legality issues)	0.0000043	**0.0104212**	0.0005390	0.0000046
"Contact Prescriber" (final contact, validity and legality issues)	0.0000025	0.0000028	0.0005436	< 1E-07
"Document Changes" (final changes, validity and legality issues)	0.0000024	< 1E-07	0.0005436	< 1E-07
"Is Appropriate and Safe?" (final check, appropriateness and safety issues)	0.0000001	0.0002864	0.0005433	0.0000003
"Contact Patient" (final contact, appropriateness and safety issues)	0.0000004	0.0000004	0.0005436	< 1E-07
"Contact Prescriber" (final contact, appropriateness and safety issues)	0.0000005	0.0000005	0.0005436	< 1E-07
"Document Changes" (final changes, appropriateness and safety issues)	< 1E-07	< 1E-07	0.0005436	< 1E-07
"Is Correct Label Attached?"	0.0000015	0.0000029	0.0005369	0.0000067
"Is Label Data Correct?"	0.0000015	0.0000027	0.0005373	0.0000063
"Is Drug Correct"	< 1E-07	< 1E-07	0.0005429	0.0000007
"Is Quantity of Drug Correct?"	0.0000709	0.0001130	0.0004509	**0.0000927**
"Is Correct Auxiliary Label Attached?"	0.0000002	0.0000004	0.0005423	0.0000013
"Give Filled Rx to Patient"	0.0001435	0.0002250	0.0003989	**0.0001447**
"Deliver Counseling to Patient"	0.0001435	NA	0.0003989	**0.0001447**

Note that bold entries represent values from each column that are orders of magnitude larger than the other entries.

tity of Drug Correct?," "Give Filled Rx to Patient," and "Deliver Counseling to Patient" are the tasks whose correction would most improve medication error rates. Thus, improving these tasks would likely have the most impact on procedure reliability. For example, if we further modify the procedure model so that the performance of these five tasks are always correct, we get a new rate of 0.0000112, a 97.94% decrease. Examining these results more closely, it is not surprising that "Print Label," "Give Filled Rx to Patient," and "Deliver Counseling to Patient" appear to have the most impact on final reliability because these three tasks are the only ones without verification steps in the remainder of the procedure. Thus, we would recommend that future research focus on enhanced technology or procedural changes that will assist in these tasks or facilitate their verification. The two other tasks with the most impact were both related to the counting or dispensing of the proper amount of medication ("Count Medication" and "Quantity of Drug Correct?"). Given that automated dispensing equipment exists (but was not considered in our evaluations) our results suggest that this could be a good investment from a reliability perspective.

Clearly, the development and application of SAFPHRx has made significant contributions to both community pharmacy practice and HRA. We discuss the implications of our findings as well as direction for future research below.

6.1. Community Pharmacy Reliability

The results we obtained in our analyses match the comparable rates we were able to obtain in the literature. However, there are limitations to the results we presented here. First, all data was modeled after a nominal community pharmacy dispensing procedure. There can definitely be variation in this process across the country and internationally. Further, assessments were conducted with a single subject matter expert with experience in community pharmacies in New York and North Carolina. More general recommendations would likely require a broader assessment base. Finally, for reasons covered in the introduction, there is limited data on which to compare our results.

It is important to note that the purpose of SAFPHRx is not to be used to make general recommendations about

community pharmacy practices. Rather, it is intended to be a technology that individual pharmacies can use to understand and improve their reliability. Thus, specific process improvements recommended in the paper should only be considered tentative. If a pharmacy wishes to apply SAFPHR℞ to an analysis of their business, they should generally follow the steps enumerated in Section 3. Practically, this would involve:

1. updating the procedure in Fig. 3 to match the one they use,
2. identifying the distribution of prescriptions filled in the pharmacy similar to what is shown in Fig. 4,
3. assessing each of the static CPCs for each task and each dynamic CPC for each time zone,
4. identifying reliability rates for non-human sources of error,
5. incorporating this information into a formal prism model using the general principles we outline,
6. formulating specifications specific to the model using our specification property patterns, and
7. using PRISM to calculate reliability rates by verifying the specification properties against the model.

Note that for our survey we used percentages of prescriptions filled in a given time zone to assess dynamic CPCs in our survey. Because our subject matter expert was familiar with these numbers and the loads experienced by community pharmacies, he was able to interpret these in terms of overall load. However, this may not be the best practice for all pharmacists, for whom the total number of prescriptions filled per zone could potentially be more intuitive.

Future research should focus on applying and evaluating SAFPHR℞ in more ecological contexts. To this end, we plan to validate SAFPHR℞ in at least two different ways. First, we will perform SAFPHR℞ analyses for different individual pharmacies and compare our results with internal data these pharmacies report. Second, we ultimately hope to perform simulation studies in a real pharmacy environment to not only validate the predictions of our observed rates, but to test the ability of SAFPHR℞ to predict the effectiveness of interventions. Beyond these validation efforts, there are enhancements to SAFPHR℞ that could improve its predictions. These are explored next.

6.2. Error Rates of Arriving Prescriptions

While our analyses showed that they had minimal impact on overall error rates, the probabilities of prescriptions arriving with validity and legality issues (P_V) or appropriateness and safety problems (P_S) were not based on documented values. Finding accurate value for these could improve method predictions, and would generally be a contribution to the pharmacy community. Thus, future work should conduct a comprehensive study to identify accurate values for these for a range of pharmacies.

6.3. SAFPHR℞ and HRA Development

SAFPHR℞ makes significant contributions to the science of human reliability. Even Eric Hollnagel, the creator of CREAM, has criticized HRAs for only focusing on human error, not considering its context as part of a dynamic system, failing to account for errors from other system elements, and ignoring error interactions (Hollnagel, 1998a). By adapting CREAM's approach to HRA for use with PRISM probabilistic model checking, SAFPHR℞ addresses all of these issues. By allowing us to model how a prescription moves through a pharmacy, probabilistic model checking accounts for the dynamism of the prescription dispensing process. Because SAFPHR℞ builds off of CREAM, it is able to provide human reliability predictions that are theoretically-grounded and well-validated. Through the synergistic use of both CREAM and model checking, SAFPHR℞ is able to account for human errors, other sources of error, and their potential interactions in a dynamic environment. Thus, SAFPHR℞ offers an unprecedented ability to predict system failure rates while accounting for human error. Despite these improvements, there are limits to SAFPHR℞. We discuss how these could be addressed in future work below.

6.3.1. Generalizability

While the contributions of SAFPHR℞ specifically target pharmacy applications, SAFPHR℞ could easily be adapted to other areas. Human error is a problem in safety critical domains beyond community pharmacies (Sujan et al., 2018; Holmberg and Kahlbom, 2019; Burns and Bonaceto, 2018; Hogenboom et al., 2018). These include other pharmacy environments, healthcare in general, aviation, automobile operation, unmanned vehicle control, and industrial environment prone to occupational accidents. Future research should investigate how SAFPHR℞ can be used in other critical domains.

6.3.2. Point Reliability Estimates

Because SAFPHR℞ makes use of basic CREAM, it is only able to produce ranges (Fig. 2) of error rate predictions (with point estimates approximated with averages). However, there are two variation of "Extended CREAM" (Hollnagel, 1998a) that are capable of producing precise predictions of error rates without the need for range averaging. Thus, in future work, we will improve the predictions of SAFPHR℞ by making it compatible with Extended CREAM (Hollnagel, 1998a) and assessing which of its variants are the most accurate.

6.3.3. Scalability

Probabilistic model checking, by virtue of being exhaustive, can have problems with scalability (Katoen, 2010). That is, as the size of the target model grows, it can take exponentially larger amounts of computer memory and computation time to check specification properties (a situation commonly called the "state explosion problem")

(Clarke et al., 1999). However, in this application, it took between 2.615 and 440.651 seconds to check each property, with an average time of 95.8 seconds. These times are very reasonable. Because this was an analysis of a full pharmacy procedure, it is unlikely that scale would present a significant problem for other pharmacy environments. It is conceivable, however, that more complex domains could present scalability issues. Fortunately, PRISM support an analysis technique called statistical model checking (Kwiatkowska et al., 2011), an approach that can be used to get approximate results in situations where scalability is a constraint. It works by creating many random traces through the formal model, evaluating the property being checked in each trace, and aggregating the results into an approximate answer. Future research should investigate the performance/accuracy trade-offs of using this technique with SAFPHRx.

6.3.4. Parameter Exploration

The current implementation of SAFPHRx requires analysts to manually explore how parameters impact analysis results. For example, the sensitivity analyses reported in Table 2 and the New Prob. Values reported in Table 4. Such a process is labor intensive for analysts. A potentially useful feature of PRISM that could automate such explorations is parametric model checking. Parametric model checking describes emerging techniques that enable analysts to explore model parameter values to find system conditions that will address discovered problems (Hahn et al., 2011b,a). Future research should explore how this feature can be incorporated into SAFPHRx in a way that will facilitate reliability improvements in community pharmacy procedures.

6.3.5. Task Modeling Improvements

The task modeling currently used in SAFPHRx is based on flow diagrams. This did not limit the scope of the analyses presented here. However, there are more-sophisticated, formal, task modeling systems such as the Enhanced Operator Function Model (EOFM) (Bolton and Bass, 2009a; Bolton et al., 2011) and EOFM with Communications (EOFMC) (Bolton and Bass, 2017; Bass et al., 2011) that offer more sophisticated task ordering, parallelism, nondeterminism, and human team communication and coordination. These task systems have also been used in formal methods analyses to generate and assess the impact of potentially unanticipated erroneous human behavior in complex systems (Bolton and Bass, 2017, 2009b, 2011; Bolton, 2010; Bolton et al., 2012; Bolton and Bass, 2013b,a; Pan and Bolton, 2018; Bolton, 2015; Bolton et al., 2019; Bolton, 2017), but without probabilistic considerations. Future work should investigate how EOFM and EOFMC technology could be incorporated into SAFPHRx to enable more sophisticated procedure modeling and error prediction in reliability computations.

6.3.6. Tool Support

The modeling techniques and specification property patterns we provide here should enable someone to perform their own safer analyses. However, pharmacist, who would be the target audience for applying this method, are likely not familiar with the formal modeling and temporal logic concepts SAFPHRx depends on. Thus, we ultimately intend to make SAFPHRx more approachable by developing a point-and-click interface that will make it easy for analysts to construct procedure models, assess CPCs and non-human error rates, and automatically generate specification properties run their associated verification analyses. Constructing and evaluating this interface will be the subject of future research.

Acknowledgments

We would like to thank Dr. Lu Feng (University of Virginia) for helping us use the PRISM model checker in preliminary analyses reported in (Zheng et al., 2017). We would also like to thank Kara Wilcox and Cody Racha (University at Buffalo) who piloted the survey we designed and provided comments to help us improve it.

References

Allan, E.A., Barker, K.N., 1990. Fundamentals of medication error research. American Journal of Health-System Pharmacy 47, 555–571.

Ashcroft, D.M., Morecroft, C., Parker, D., Noyce, P.R., 2006. Likelihood of reporting adverse events in community pharmacy: An experimental study. Quality and Safety in Health Care 15, 48–52.

Bass, E.J., Bolton, M.L., Feigh, K., Griffith, D., Gunter, E., Mansky, W., Rushby, J., 2011. Toward a multi-method approach to formalizing human-automation interaction and human-human communications, in: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, IEEE, Piscataway. pp. 1817–1824.

Bates, D.W., 2007. Preventing medication errors: a summary. American Journal of Health-System Pharmacy 64, S3–S9.

Bell, J., Holroyd, J., 2009. Review of human reliability assessment methods. Technical Report RR679. Health and Safety Executive. Derbyshire.

Berdot, S., Gillaizeau, F., Caruba, T., Prognon, P., Durieux, P., Sabatier, B., 2013. Drug administration errors in hospital inpatients: a systematic review. PloS one 8, e68856.

Bolton, M.L., 2010. Using Task Analytic Behavior Modeling, Erroneous Human Behavior Generation, and Formal Methods to Evaluate the Role of Human-automation Interaction in System Failure. Ph.D. thesis. University of Virginia. Charlottesville.

Bolton, M.L., 2015. Model checking human–human communication protocols using task models and miscommunication generation. Journal of Aerospace Information Systems 12, 476–489.

Bolton, M.L., 2017. A task-based taxonomy of erroneous human behavior. International Journal of Human-Computer Studies 108, 105–121.

Bolton, M.L., Bass, E.J., 2009a. Enhanced operator function model: A generic human task behavior modeling language, in: Proceedings of the IEEE International Conference on Systems Man and Cybernetics, IEEE, Piscataway. pp. 2983–2990.

Bolton, M.L., Bass, E.J., 2009b. A method for the formal verification of human interactive systems, in: Proceedings of the 53rd Annual Meeting of the Human Factors and Ergonomics Society, HFES, Santa Monica. pp. 764–768.

Bolton, M.L., Bass, E.J., 2011. Evaluating human-automation interaction using task analytic behavior models, strategic knowledge-based erroneous human behavior generation, and model checking, in: Proceedings of the IEEE International Conference on Systems Man and Cybernetics, IEEE, Piscataway. pp. 1788–1794.

Bolton, M.L., Bass, E.J., 2013a. Evaluating human-human communication protocols with miscommunication generation and model checking, in: Proceedings of the Fifth NASA Formal Methods Symposium. Moffett Field: NASA Ames Research Center, NASA Ames Research Center, Moffett Field. pp. 48–62.

Bolton, M.L., Bass, E.J., 2013b. Generating erroneous human behavior from strategic knowledge in task models and evaluating its impact on system safety with model checking. IEEE Transactions on Systems, Man and Cybernetics: Systems 43, 1314–1327.

Bolton, M.L., Bass, E.J., 2017. Enhanced operator function model (EOFM): A task analytic modeling formalism for including human behavior in the verification of complex systems, in: Weyers, B., Bowen, J., Dix, A., Palanque, P. (Eds.), The Handbook of Formal Methods in Human-Computer Interaction. Springer, Cham, pp. 343–377.

Bolton, M.L., Bass, E.J., Siminiceanu, R.I., 2012. Generating phenotypical erroneous human behavior to evaluate human-automation interaction using model checking. International Journal of Human-Computer Studies 70, 888–906.

Bolton, M.L., Bass, E.J., Siminiceanu, R.I., 2013. Using formal verification to evaluate human-automation interaction in safety critical systems, a review. IEEE Transactions on Systems, Man and Cybernetics: Systems 43, 488–503.

Bolton, M.L., Molinaro, K.A., Houser, A.M., 2019. A formal method for assessing the impact of task-based erroneous human behavior on system safety. Reliability Engineering & System Safety 188, 168–180.

Bolton, M.L., Siminiceanu, R.I., Bass, E.J., 2011. A systematic approach to model checking human-automation interaction using task-analytic models. IEEE Transactions on Systems, Man, and Cybernetics, Part A 41, 961–976.

Boring, R.L., Dudenhoeffer, D.D., Hallbert, B.P., Gore, B.F., 2006. Modeling human reliability analysis using MIDAS. Technical Report. Idaho National Laboratory (INL).

Burns, K., Bonaceto, C., 2018. An empirically benchmarked human reliability analysis of general aviation. Reliability Engineering & System Safety .

Bye, A., Hollnagel, E., Brendeford, T.S., 1999. Human–machine function allocation: a functional modelling approach. Reliability Engineering & System Safety 64, 291–300.

Castiglia, F., Giardina, M., Caravello, F., 2008. Fuzzy fault tree analysis in modern γ-ray industrial irradiator: Use of fuzzy version of heart and cream techniques for human error evaluation, in: 9th International Conference on Probabilistic Safety Assessment and Management, PSAM9, Kao, T.. pp. 1377–1384.

Clarke, E.M., Grumberg, O., Peled, D.A., 1999. Model Checking. MIT Press, Cambridge.

Clemens, P.L., Simmons, R.J., 1998. System Safety and Risk Management: NIOSH Instructional Module. U.S. Department of Health and Human Services, Cincinnati.

Commission, U.N.R., et al., 2000. Technical basis and implementation guidelines for a technique for human event analysis (ATHEANA). NUREG-1624, Rev 1.

Cooper, S., Ramey-Smith, A., Wreathall, J., Parry, G., Bley, D., Luckas, W., Taylor, J., Barriere, M., 1996. A technique for human event analysis (ATHEANA)-technical basis and methodological description. Technical Report. Report No. NUREG/CR-6350, US Nuclear Regulatory Commission, Brookhaven National Laboratory, Upton, New York.

De Felice, F., Petrillo, A., Carlomusto, A., Ramondo, A., 2012. Human reliability analysis: A review of the state of the art. IRACST–International Journal of Research in Management & Technology (IJRMT) 2.

Deeter, J., Rantanen, E., 2012. Human reliability analysis in healthcare, in: Proceedings of Symposium on Human Factors and Ergonomics in Health Care, pp. 45–51.

Di Pasquale, V., Iannone, R., Miranda, S., Riemma, S., 2013. An overview of human reliability analysis techniques in manufacturing operations, in: Operations management. IntechOpen, pp. 221–240.

Feng, L., Humphrey, L., Lee, I., Topcu, U., 2016a. Human-interpretable diagnostic information for robotic planning systems, in: International Conference on Intelligent Robots and Systems, IEEE, Piscataway. pp. 1673–1680.

Feng, L., Wiltsche, C., Humphrey, L., Topcu, U., 2016b. Synthesis of human-in-the-loop control protocols for autonomous systems. IEEE Transactions on Automation Science and Engineering 13, 450–462.

Flynn, E.A., Barker, K.N., Carnahan, B.J., 2003. National observational study of prescription dispensing accuracy and safety in 50 pharmacies. Journal of the American Pharmaceutical Association 43, 191–200.

Fujita, Y., 1992. Human reliability analysis: a human point of view. Reliability Engineering & System Safety 38, 71–79.

Fujita, Y., Hollnagel, E., 2004. Failures without errors: quantification of context in HRA. Reliability Engineering & System Safety 83, 145–151.

Geng, J., Murè, S., Baldissone, G., Camuncoli, G., Demichela, M., 2015. Human error probability estimation in ATEX-HMI area classification: From THERP to FUZZY CREAM. Chemical Engineering Transactions 43, 1243–1248.

Gilligan, A.M., Miller, K., Mohney, A., Montenegro, C., Schwarz, J., Warholak, T.L., 2012. Analysis of pharmacists? interventions on electronic versus traditional prescriptions in 2 community pharmacies. Research in social and administrative pharmacy 8, 523–532.

Hahn, E.M., Han, T., Zhang, L., 2011a. Synthesis for PCTL in parametric Markov decision processes, in: NASA Formal Methods Symposium, Springer, Berlin. pp. 146–161.

Hahn, E.M., Hermanns, H., Zhang, L., 2011b. Probabilistic reachability for parametric Markov models. International Journal on Software Tools for Technology Transfer 13, 3–19.

Hannaman, G., Spurgin, A., Lukic, Y., 1984. Human cognitive reliability model for PRA analysis. Technical Report NUS-4531. Palo Alto.

Hogenboom, S., Rokseth, B., Vinnem, J.E., Utne, I.B., 2018. Human reliability and the impact of control function allocation in the design of dynamic positioning systems. Reliability Engineering & System Safety .

Hollnagel, E., . CREAM - Cognitive Reliability and Error Analysis Method. URL: `http://erikhollnagel.com/ideas/cream.html`.

Hollnagel, E., 1998a. Cognitive Reliability and Error Analysis Method (CREAM). Elsevier, Oxford.

Hollnagel, E., 1998b. Context, cognition and control, in: Waern, Y. (Ed.), Co-operative Process Management, Cognition and Information Technology. Taylor & Francis, London, pp. 27–52.

Hollnagel, E., 2017. FRAM: the functional resonance analysis method: modelling complex socio-technical systems. CRC Press.

Hollnagel, E., Kaarstad, M., Lee, H.C., 1999. Error mode prediction. Ergonomics 42, 1457–1471.

Holmberg, J.E., Kahlbom, U., 2019. Application of human reliability analysis in the deterministic safety analysis for nuclear power plants. Reliability Engineering & System Safety .

IOM, 2006. Preventing Medication Errors. The National Academies Press, Washington, DC.

Katoen, J.P., 2010. Advances in probabilistic model checking, in: International Workshop on Verification, Model Checking, and Abstract Interpretation, Springer, Berlin. pp. 25–25.

Kaushal, R., Bates, D.W., Landrigan, C., McKenna, K.J., Clapp, M.D., Federico, F., Goldmann, D.A., 2001. Medication errors and adverse drug events in pediatric inpatients. Jama 285, 2114–2120.

Kim, J.W., Jung, W., 2003. A taxonomy of performance influencing factors for human reliability analysis of emergency tasks. Journal of Loss Prevention in the Process Industries 16, 479–495.

Kim, M.C., Seong, P.H., Hollnagel, E., 2006. A probabilistic approach for determining the control mode in CREAM. Reliability Engineering & System Safety 91, 191–199.

Kirwan, B., Gibson, H., Kennedy, R., Edmunds, J., Cooksley, G., Umbers, I., 2004. Nuclear action reliability assessment (NARA): A data-based HRA tool, in: International Conference on Probabilistic Safety Assessment and Management, Springer, Berlin. pp. 1206–1211.

Kwiatkowska, M., Norman, G., Parker, D., 2007. Stochastic model checking, in: Bernardo, M., Hillston, J. (Eds.), Formal Methods for the Design of Computer, Communication and Software Systems: Performance Evaluation, Springer, Berlin. pp. 220–270.

Kwiatkowska, M., Norman, G., Parker, D., 2011. PRISM 4.0: Verification of probabilistic real-time systems, in: International Conference on Computer Aided Verification, Springer. pp. 585–591.

Lao, K.S., Chui, C.S., Man, K.K., Lau, W.C., Chan, E.W., Wong, I.C., 2016. Medication safety research by observational study design. International journal of clinical pharmacy 38, 676–684.

Lee, S.W., Kim, A.R., Ha, J.S., Seong, P.H., 2011. Development of a qualitative evaluation framework for performance shaping factors (PSFs) in advanced MCR HRA. Annals of Nuclear Energy 38, 1751–1759.

Li, Y., Mosleh, A., 2019. Dynamic simulation of knowledge based reasoning of nuclear power plant operator in accident conditions: modeling and simulation foundations. Safety Science 119, 315–329.

Liao, H., Forester, J., Dang, V.N., Bye, A., Chang, Y.H.J., Lois, E., 2019. Assessment of hra method predictions against operating crew performance: part iii: conclusions and achievements. Reliability Engineering & System Safety , 106511.

Liu, P., Qiu, Y., Hu, J., Tong, J., Zhao, J., Li, Z., 2018. Expert judgments for performance shaping factors' multiplier design in human reliability analysis. Reliability Engineering & System Safety .

Makary, M.A., Daniel, M., 2016. Medical error? the third leading cause of death in the us. Bmj 353, i2139.

Odukoya, O., Schleiden, L., Chui, M., 2015. The hidden role of community pharmacy technicians in ensuring patient safety with the use of e-prescribing. Pharmacy 3, 330–343.

Pan, D., Bolton, M.L., 2018. Properties for formally assessing the performance level of human-human collaborative procedures with miscommunications and erroneous human behavior. International Journal of Industrial Ergonomics 63, 75–88.

Pape, T.M., 2001. Searching for the final answer: factors contributing to medication administration errors. The Journal of Continuing Education in Nursing 32, 152–160.

Parker, D., Norman, G., Kwiatkowska, M., 2017. PRISM Manual. Technical Report. University of Oxford. URL: http://www.prismmodelchecker.org/manual/.

Parker, J. (Ed.), 2013. Meeting the Joint Commission's 2013 National Patient Safety Goals. Oakbrook Terrace. URL: http://www.healtheast.org/images/stories/patient-safety/2014_jc_patient_safety_goals.pdf.

Rantanen, E., Deeter, J., . Human reliability analysis in healthcare (technical report no. rit/psy/tr-11/1). URL: https://scholarworks.rit.edu/article/1428/.

Rantanen, E., Deeter, J., Burke, S., Wang, Y., 2012. Analysis of hospital oncology department pharmacy operations, in: Proceedings of Symposium on Human Factors and Ergonomics in Health Care, pp. 27–34.

Reer, B., 2008. Review of advances in human reliability analysis of errors of commission part 2: EOC quantification. Reliability Engineering & System Safety 93, 1105–1122.

Sujan, M.A., Embrey, D., Huang, H., 2018. On the application of human reliability analysis in healthcare: opportunities and challenges. Reliability Engineering & System Safety .

Swain, A., 1987. Accident Sequence Evaluation Program Human Reliability Analysis Procedure. Technical Report NUREG/CR-4772. US Nuclear Regulatory Commission. Washington, DC.

Swain, A.D., 1990. Human reliability analysis: Need, status, trends and limitations. Reliability Engineering & System Safety 29, 301–313.

Swain, A.D., Guttmann, H.E., 1983. Handbook of human-reliability analysis with emphasis on nuclear power plant applications: Final report. Technical Report 7011944. Sandia National Labs. Albuquerque.

Szeinbach, S., Seoane-Vazquez, E., Parekh, A., Herderick, M., 2007. Dispensing errors in community pharmacy: perceived influence of sociotechnical factors. International Journal for Quality in Health Care 19, 203–209.

The Joint Commission, 2014. 2014 Hospital National Patient Safety Goals. Technical Report. The Joint Commission. URL: http://www.healtheast.org/images/stories/patient-safety/2014_jc_patient_safety_goals.pdf.

The Joint Commission, 2015. 2015 Hospital National Patient Safety Goals. Technical Report. The Joint Commission. URL: http://www.jointcommission.org/assets/1/6/2015_HAP_NPSG_ER.pdf.

The Joint Commission, 2016a. 2016 Hospital National Patient Safety Goals. Technical Report. The Joint Commission. URL: https://www.jointcommission.org/assets/1/6/2016_NPSG_HAP_ER.pdf.

The Joint Commission, 2016b. National Patient Safety Goals Effective January 2017. Technical Report. The Joint Commission. URL: https://www.jointcommission.org/assets/1/6/NPSG_Chapter_HAP_Jan2017.pdf.

Weyers, B., Bowen, J., Dix, A., Palanque, P. (Eds.), 2017. The Handbook of Formal Methods in Human-Computer Interaction. Springer, Berlin.

Williams, J., 1986. Heart–a proposed method for assessing and reducing human error, in: 9th Advances in Reliability Technology Symposium, University of Bradford.

Williams, J., 1988. A data-based method for assessing and reducing human error to improve operational performance, in: Human Factors and Power Plants, 1988., Conference Record for 1988 IEEE Fourth Conference on, IEEE. pp. 436–450.

Wilson, D.G., McArtney, R., Newcombe, R.G., McArtney, R., Gracie, J., Kirk, C., Stuart, A., 1998. Medication errors in paediatric practice: insights from a continuous quality improvement approach. European journal of pediatrics 157, 769–774.

Wing, J.M., 1990. A specifier's introduction to formal methods. Computer 23, 8, 10–22, 24.

Witte, D., Dundes, L., 2007. Prescription for error: process defects in a community retail pharmacy. Journal of Patient Safety 3, 190–194.

Wreathall, J., 1982. Operator action trees, an approach to quantifying operator error probability during accident sequences. NUS report 4159.

Xie, H.w., Sun, Z.q., Li, X.x., LI, Z.y., Zhang, M., SHI, X.j., LI, L., 2007. An overview of typical methods for human reliability analysis. Journal-National University of Defense Technology 29, 101.

Zhao, Y., Smidts, C., 2019. A method for systematically developing the knowledge base of reactor operators in nuclear power plants to support cognitive modeling of operator performance. Reliability Engineering & System Safety 186, 64–77.

Zheng, X., Bolton, M.L., Daly, C., Feng, L., 2017. A formal human reliability analysis of a community pharmacy dispensing procedure, in: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, SAGE Publications Sage CA: Los Angeles, CA. pp. 728–732.