Matthew L. Bolton et al.: Formal Modeling of Erroneous Human Behavior and its Implications for Model Checking, Proceedings of The Sixth NASA Langley Formal Methods Workshop, p.62–64

# Formal Modeling of Erroneous Human Behavior and its Implications for Model Checking

*Matthew L. Bolton, Ellen J. Bass*

*University of Virginia, Department of Systems and Information Engineering* ,
*Charlottesville, Virginia 22904, USA*
Mlb4b@Virginia.edu Ejb4n@Virginia.edu

## Extended Abstract

Modern, safety-critical systems are inherently complex as multiple interacting subsystems and people (operators, maintenance crews, etc.) attempt to achieve multiple, often conflicting, goals. While the majority of the sub-systems (including the humanmachine interfaces to control them) are well engineered, system failures still occur: airplane crashes, air-traffic conflicts, power plant failures, defense system false alarms, etc. [1]. Such failures are often due not to the breakdown of a single component, but to a series of minor events that occur at separate times, ultimately leading to dangerous outcomes. Further, more of the pre-cursor events that lead to such outcomes are the result of human error (the error resulting from the interaction between human operators and the system) rather than equipment or component failure [2].

Formal methods, and particularly model checking, have proven useful in detecting design errors that produce system failure in computer hardware and software systems. A number of techniques also exist for modeling human behavior using formal computational structure such as Goals, Operators, Methods, and Selection rules (GOMS) [4], ConcurTaskTrees (CTT) [5], and the Operator Function Model (OFM) [6]. In addition, efforts have also been made to classify human error based on its formal characteristics. While there are a number of reasons why humans may perform an erroneous act (a sequence of activities that do not produce the intended result during human-system interaction), there are very limited formal characteristics for the way that errors can manifest themselves [2]. To address this, Hollnagel [7] classified human error based on a hierarchy of phenotypes, the formal characteristics of observable erroneous behavior. Hollnagel showed that all human errors were composed of one or more of the following errors (all observable for a single act): premature start of an action, delayed start of an action, premature finishing of an action, delayed finishing of an action, omitting an action, skipping an action, reperforming a previously performed action, repeating an action, and performing an unplanned action (an intrusion).

A variety of work has investigated the use of formal system and human behavior models in order to predict and model human error (an overview can be found in [3]). However, the majority of this work has focused on discovering mode confusion and automation surprise (preconditions for a subset of human errors), or have relied on human factors experts to incorporate erroneous behavior into human-behavior models. None of these methods have integrated model checking, human behavior modeling, and human error phenotype classification to automatically model erroneous behavior and use it to predict its contribution to system failure.

To address this, we are developing an extension of the model checking verification process [3] (Fig. 1). This framework includes three automatic processes: human error prediction, translation,

and model checking. The human error prediction process examines a normative human behavior model and a human-system interface model in order to determine what erroneous human behavior patterns are likely. It produces a modified version of the human behavior model with both the normative and erroneous behavior. The translation process uses the system model and the modified human behavior model and produces a single model that is readable by the model checker. The model checking process verifies that the system properties from the specification are true in the system model. If verification fails, the process will generate a counterexample showing how the failure condition occurred. This framework has been instantiated using the SMART and SAL modeling checking programs and used successfully with a simplified model of the Therac-25, a piece of radiological medical equipment for which human error played an important role in a fatal system failure (see [3]). The work discussed here focuses on the human error prediction process, the erroneous human behavior model it produces, and its implication for model checking.

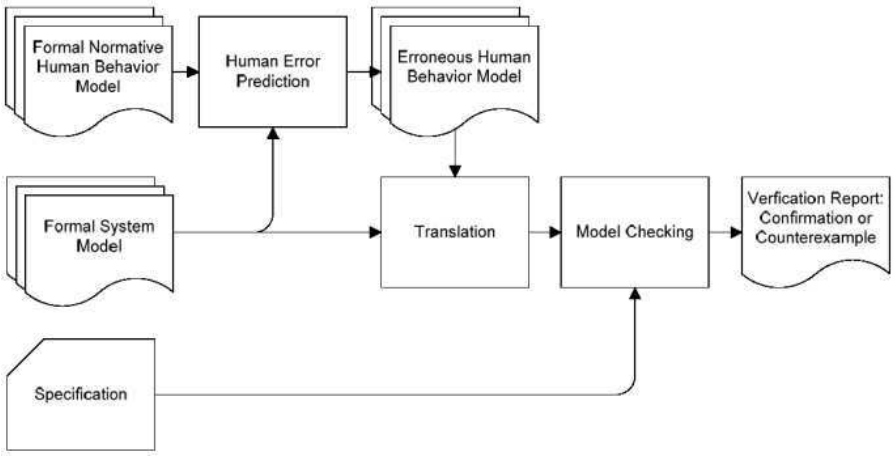


Figure 1. Human error and system failure prediction framework

This work discusses a systematic means of modifying a normative human behavior model specified in the OFM paradigm (decomposing higher level activities into atomic actions) in order to incorporate the observable erroneous behavior identified by Hollnagel (all of which can be constructed from errors at the atomic action level). Given the hierarchal nature of the OFMs and Hollnagel s error phenotypes, this process can be done by replacing each of an OFM s atomic actions with a set of erroneous acts that may occur at that action. Given that the framework being employed in this work assumes a formal model of the human-computer interface and full normative human behavior models, both can be used to determine which of Hollnagel s phenotypes can manifest themselves at a particular action.

Human behavior models used with the proposed framework (Fig. 1) have two important implications for model checking. First, given the nature of model checking, any system containing human-system interaction that is evaluated via model checking will encompass a superset of human behavior beyond what is likely. In this context, the erroneous behavior model can be viewed as a filter for the system model as it limits the human behavior possibilities the model checker needs to evaluate. Thus, we may be able to reduce the system model s state space during the translation process in Fig. 1, potentially alleviating the state explosion problem. Second, the behavior models can be used to explain how human error may have contributed to a system failure identified in a counterexample. This is useful as it may suggest interface or other design changes that prevent the error from occurring.

## 1. References

[1] Perrow, C., Normal Accidents: Living with High-Risk Technologies, Basic Books, New York (1984)

[2] Reason, J., Human Error, Cambridge University Press, Cambridge, England (1990)

[3] Bolton, M.L., Bass, E.J., Siminiceanu, R.I., Using Formal Methods to Predict Human Error and System Failures, In: Applied Human Factors and Ergonomics International 2008 (In Press)

[4] Kieras, D., John, B., The GOMS family of analysis techniques: tools for design and evaluation, CMU-HCII-94-106 (1994)

[5] Mori, G., Patern, F., Santoro, C., CTTE: Support for Developing and Analyzing Task Models for Interactive System Design, IEEE Transactions on Software Engineering, 28, 8, 797–813 (2002)

[6] Thurman, D.A., Chappell, A.R., Mitchell, C.M., An enhanced architecture for OFMspert: A Domain-independent System for Intent Inferencing. IEEE International Conference on Systems Man and Cybernetics, pp. 955–960. IEEE Press, New York (1998)

[7] Hollnagel, E., 1993, The Phenotype of Erroneous Actions, International Journal of Man-Machine Studies, 39, 1–32 (1993).