Model Checking Human-automation Interaction with Enhanced Operator Function Model

Matthew L. Bolton
San José State University
Research Foundation
NASA Ames Research Center
Moffett Field, CA USA
matthew.l.bolton@nasa.gov

Ellen J. Bass
Department of Systems and Information Engineering
University of Virginia
Charlottesville, VA USA
ejb4n@virginia.edu

ORIGIN AND UNDERLYING PRINCIPLES

Engineers use task analytic behavior models to describe the normative human behaviors required to control a system [12]. These models represent the mental and physical activities operators use to achieve the goals that the system was designed to support. Enhanced Operator Function Model (EOFM) [9], an extension of the Operator Function Model [13], represents human behavior as an input-output system using an XML notation.

An instantiated EOFM describes inputs from external sources and how a human operator produces actions as part of a task. Tasks in EOFMs are hierarchical representations of goal-driven activities that decompose into lower level activities, and, finally, atomic actions. EOFMs express task knowledge by explicitly specifying the conditions under which human operator activities can execute (preconditions), when they can repeat (repeat conditions), and what must be true when they finish (completion conditions) in Boolean expressions. Any activity can decompose into one or more activities or actions (sub-acts). A decomposition operator specifies the temporal relationships between and the cardinality of the decomposed sub-acts (when they can execute relative to each other and how many can execute). EOFM supports all of the decomposition operators in Table 1. EOFM also supports a visual notation (see Figure 1 for an example).

EOFM has formal semantics that specify how an instantiated EOFM model executes [9]. Specifically, each activity or action can have one of three execution states: waiting to execute (*Ready*), executing (*Executing*), and done (*Done*). An activity or action transitions between each of these states based on its current state; the state of its immediate parent, its siblings (activities or actions contained in the same decomposition), and its immediate children in the hierarchy; and the decomposition operators that connect the activity to its parent and its children. This formal semantics allows an instantiated EOFM to be automatically translated into a formal model [9] capable of being evaluated by a model checker, the Symbolic Analysis Laboratory (SAL) [10] in our case.

Operator	Description
optor_seq	Zero or more of the sub-acts must execute in any order one at a time.
optor_par	Zero or more of the sub-acts must execute in any order and can execute in parallel.
or_seq	One or more of the sub-acts must execute in any order one at a time.
or_par	One or more of the sub-acts must execute in any order and can execute in parallel.
and_seq	All of the sub-acts must execute in any order one at a time.
and_par	All of the sub-acts must execute in any order and can execute in parallel.
xor	Exactly one sub-act must execute.
ord	All sub-acts must execute in the order they appear.
sync	All sub-acts must execute synchronously.

Table 1. EOFM Decomposition Operators

MODELED RELATIONSHIPS

EOFMs are typically used to represent normative human behavior. However, it is possible to generate potentially unanticipated erroneous human behavior in instantiated EOFMs and thus include it in formal system models.

Erroneous behavior can be produced in two different ways. In the first [5], each action in an EOFM task is replaced with a generative that allows for the performance of Hollnagel's [11] zero-order phenotypes of erroneous action. Through multiple performances of zero-order phenotypes, more complicated erroneous behaviors are possible. In the second erroneous behavior generation technique [7], the

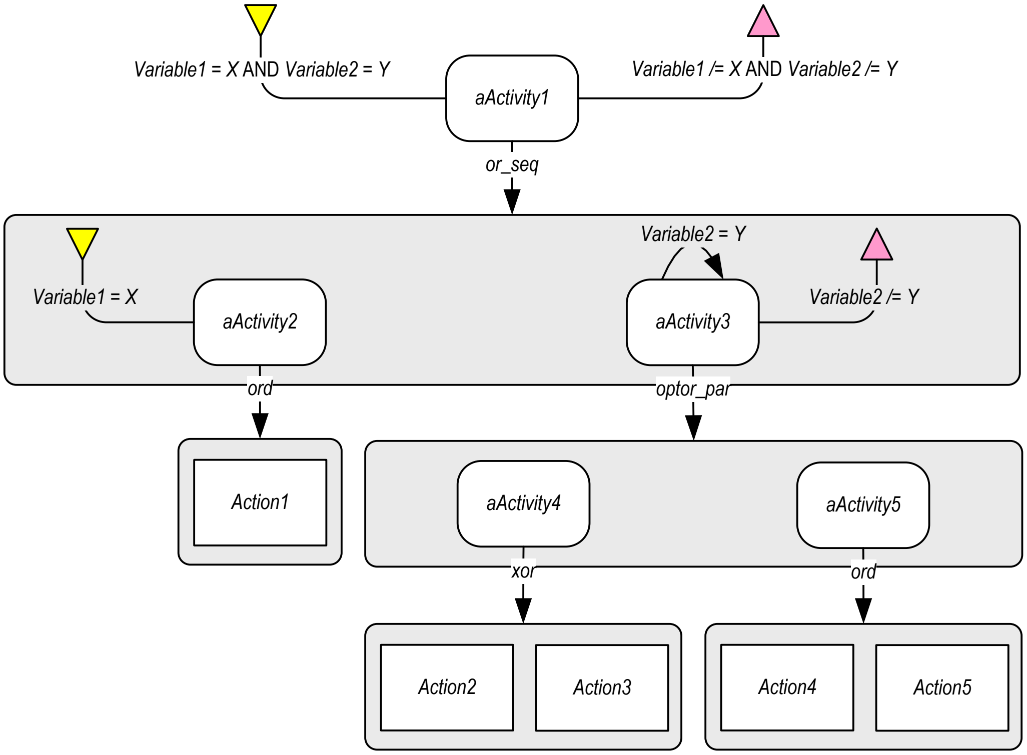

Figure 1. The visual representation of a task structure in an instantiation of an EOFM. EOFMs can be represented visually as tree-like graphs. Actions are rectangles and activities are rounded rectangles. An activity's decomposition is presented as an arrow, labeled with the decomposition operator, that points to a large rounded rectangle containing the decomposed activities or actions. Conditions on activities are represented as shapes or arrows (annotated with the logic) connected to the activity that they constrain. A precondition is a yellow, downward-pointing triangle; a completion condition is a magenta, upward-pointing triangle; and a repeat condition is an arrow recursively pointing to the top of the activity.

formal semantics of EOFM are extended in order to model human operator attention failures (Reason's [14] slips) that enable activities to be erroneously omitted, repeated, or committed. In both cases, a maximum is used to control the number of erroneous behaviors considered in a given evaluation. Both of these erroneous behavior generation techniques have been implemented as options in the EOFM to SAL translator.

Translated EOFM instances fit into a larger formal modeling architectural framework that supports concepts important to human-automation interaction. This encompasses models of human missions (i.e. goals), human task behavior (the translated EOFM instance), human-device interfaces (displays and controls available to the human operator), device automation (underlying device behavior), and the operational environment [4].

PROBLEMS ADDRESSED

The formal nature of the framework allows models created with it to be evaluated with a model checker. Model checking is an automated formal verification process that exhaustively searches a system's statespace to see if it can find a violation of specification properties. If no violation is found, the model checker has proven that the model adheres to the specification. Otherwise, the model checker produces a counterexample that illustrates how a violation occurred.

Thus, a model checker can be used on formal models that contain translated EOFM instances to prove whether or not the modeled human behavior, and the resulting human-automation interaction, will contribute to a violation of system safety (encoded in a specification property). This can include any generated (and thus potentially unanticipated) erroneous human behavior. System safety has been evaluated with normative human task behavior [9], normative human performance of checklist procedures [8], generated phenotypical erroneous human behavior [5], and generated erroneous human behavior caused by failures of attention [7]. It is also possible to use the visual notation of the EOFM to help analysts diagnose specification violations reported in counterexamples [6].

APPLICATIONS

EOFM has been used with model checking to find problems in, and explore design interventions for a number of applications. These include a Patient Controlled Analgesia Pump [4, 3, 7], an automobile with a simple cruise control [9], an aircraft instrument landing checklist procedure [8], and a radiation therapy machine [5].

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

There are a number of potential development opportunities for EOFM and its associated analyses. Firstly, analyses that utilize EOFMs, especially those with erroneous human behavior generation, do not scale well [2]. Thus current work is investigating ways of improving the EOFM to SAL translation in order to improve scalability. Secondly, the presented method has almost exclusively been used to evaluate single operator systems. However, ongoing research is investigating how to model human communication and coordination with EOFM [1]. Thirdly, in order to assist analysts in evaluating human-automation interaction with EOFM, work is currently investigating how specification properties indicative of good human-automation interaction can be generated automatically from instantiated EOFMs and used in formal verification analyses [3]. Finally, EOFM currently does not integrate with other infrastructures designed to evaluate human-automation interaction formally (see [2]). Future work should attempt to rectify this limitation.

ACKNOWLEDGEMENT

The project described was supported in part by Grant Number T15LM009462 from the National Library of Medicine (NLM), NASA Cooperative Agreement NCC1002043, and NASA award NNA10DE79C. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIA, NASA, the NLM, or the National Institutes of Health.

REFERENCES

1. Bass, E. J., Bolton, M. L., Feigh, K., Griffith, D., Gunter, E., Mansky, W., and Rushby, J. 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, 2011), 1817–1824.

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

3. Bolton, M. L. Validating human-device interfaces with model checking and temporal logic properties automatically generated from task analytic models. In *Proceedings of the 20th Behavior Representation in Modeling and Simulation Conference*, The BRIMS Society (Sundance, 2011), 130–137.

4. Bolton, M. L., and Bass, E. J. Formally verifying human-automation interaction as part of a system model: Limitations and tradeoffs. Innovations in *Systems and Software Engineering: A NASA Journal 6*, 3 (2010), 219–231.

5. Bolton, M. L., and Bass, E. J. Using task analytic models and phenotypes of erroneous human behavior to discover system failures using model checking. In *Proceedings of the 54th Annual Meeting of the Human Factors and Ergonomics Society*, Human Factors and Ergonomics Society (Santa Monica, 2010), 992–996.

6. Bolton, M. L., and Bass, E. J. Using task analytic models to visualize model checker counterexamples. In *Proceedings of the 2010 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE (Piscataway, 2010), 2069–2074.

7. Bolton, M. L., and Bass, E. J. 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, 2011). in press.

8. Bolton, M. L., and Bass, E. J. Using model checking to explore checklist-guided pilot behavior. *International Journal of Aviation Psychology* (In Press).

9. Bolton, M. L., Siminiceanu, R. I., and Bass, E. J. A systematic approach to model checking human-automation interaction using task-analytic models. *IEEE Transactions on Systems, Man, and Cybernetics, Part A 41*, 5 (2011), 961–976.

10. De Moura, L., Owre, S., and Shankar, N. The SAL language manual. Tech. Rep. CSL-01-01, Computer Science Laboratory, SRI International, Menlo Park, 2003.

11. Hollnagel, E. The phenotype of erroneous actions. *International Journal of Man-Machine Studies 39*, 1 (1993), 1–32.

12. Kirwan, B., and Ainsworth, L. K. *A Guide to Task Analysis*. Taylor and Francis, London, 1992.

13. Mitchell, C. M., and Miller, R. A. A discrete control model of operator function: A methodology for information display design. *IEEE Transactions on Systems Man Cybernetics Part A: Systems and Humans 16*, 3 (1986), 343–357.

14. Reason, J. *Human Error*. Cambridge University Press, New York, 1990.