UIVerify - a Web-Based Tool for Verification and Automatic Generation of User Interfaces

Smadar Shiffman
Asaf Degani
NASA Ames Research Center

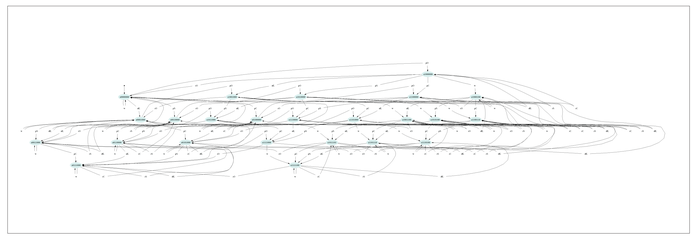
Michael Heymann
Israel Institute of Technology

Verification

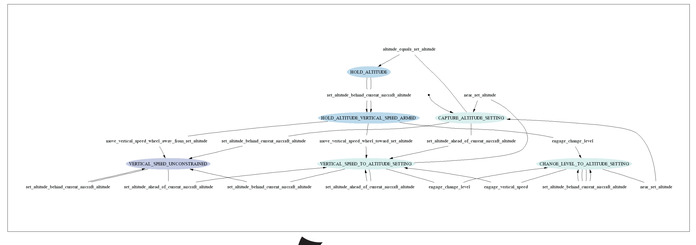
Automatic Flight Control Example

Load and confirm input

View Machine Model Graph



View User Model Graph



Obtain Results

Software tools | Interface verification tool | Interface verification input

Verification Results

error states:

At corresponding machine-model-state *VERTICAL_SPEED_TO_ALTITUDE_SETTING* and interface-model-state *VERTICAL_SPEED_UNCONSTRAINED*, specification classes differ: *armedForCapture* and *unarmedForCapture*.

Verification: Behind the scene

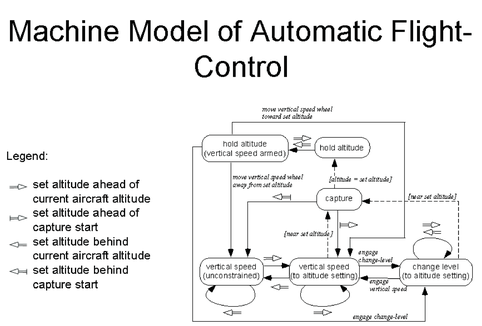
Machine-model events	User-model events
• near set altitude	• near set altitude
• set altitude *ahead* of current aircraft altitude	• set altitude *ahead* of current aircraft altitude
• set altitude *behind* current aircraft altitude	• set altitude *behind* current aircraft altitude
• set altitude *behind* capture start	• set altitude *behind* current aircraft altitude
• set altitude *ahead* of capture start	• set altitude *behind* current aircraft altitude
• set altitude *ahead* of capture start	• set altitude *ahead* of current aircraft altitude

Event mapping

Abstract

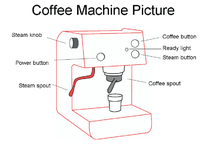
In this poster, we describe a web-based tool for verification and automatic generation of user interfaces. The verification component of the tool accepts as input a model of a machine and a model of its interface, and checks that the interface is adequate (correct). The generation component of the tool accepts a model of a given machine and the user's task, and then generates a correct and succinct interface. The poster includes two more examples of using the tool: verification of the interface to an espresso machine, and automatic generation of a succinct interface to a large hypothetical machine.

Machine Model of Automatic Flight-Control

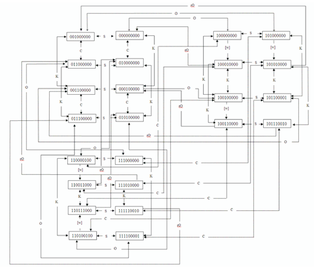


Model for Coffee Machine

Coffee Machine Picture

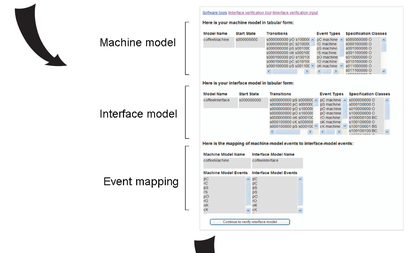


Bit no. - Representation
1 - power button
2 - coffee button
3 - steam button
4 - ready light
5 - steam knob
6 - coffee flows out of coffee spout
7 - water flows out of coffee spout
8 - steam releases out of steam spout
9 - water flows out of steam spout



Coffee Machine Example

Load and Confirm Input



Obtain results

Input machine is either non deterministic or faulty.

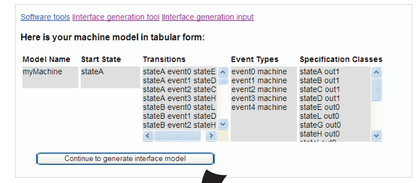
Failed to verify interface model.

Please check the input machine.

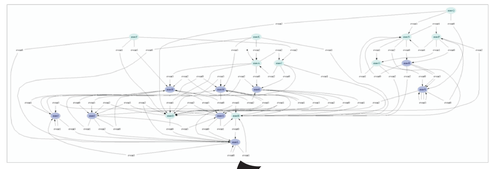
Generation

Large Machine Example

Load and confirm input

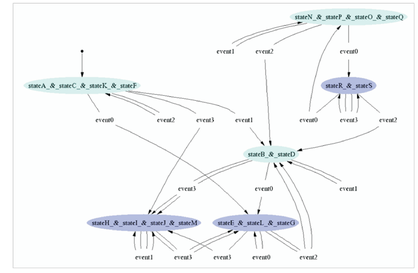


View Machine Model Graph



Obtain Results

View User Model Graph



Related Research and Conclusion

Several researchers demonstrated how formal methods can be used for analyzing user interfaces and identifying design deficiencies [1-3]. Rushby, et al., use model checking techniques to perform an iterative search for inconsistencies within a combined rule-based representation of machine and user models, and allow alteration of either the machine or the user model in-between iterations. The approach proposed by Degani and Heymann [4], which is implemented in UIVerify, employs separate descriptions for the machine and the interface, focuses on the synchronization between the two concurrent models, and provides criteria (error state, restricting state, augmenting states) for verification. The UIVerify tool is applicable for user interfaces that have many discrete modes. As for control systems, where it is important to also take into account continuous variables (such as time, speed, flight path angle, etc.), it is possible in most cases to use the methods described in [5] so as to convert the (hybrid) system into a finite state machine representation and then use UIVerify to perform the verification. Regarding the topic of automatically generating correct and succinct interfaces, the method of Heymann and Degani [6] and its implementation in UIVerify is unique in its capability to determine the simplest interface possible.

References

1. Palanque, P. and Bastide, R., (1994). Petri-net based design of user-driven interfaces using the interactive cooperative objects formalism, in Proceedings of Design, Specification and Verification of Interactive Systems, Springer Verlag, pp. 383-400.
2. Rushby, J., Using model checking to help discover mode confusions and other automation surprises, in Proceedings of the Workshop on Human Error, Safety, and System Development (HESSD), Li`ege, Belgium, 1999.
3. Doherty, G., Campos, J. C. and Harrison, M.D., Representational reasoning and verification, Formal Aspects of Computing, No. 3, 2000, pp. 1-23.
4. Degani, A. and Heymann, M., (2002). Formal verification of human-automation interaction. Human Factors, Vol. 44.1, pp 28-43.
5. Oishi M., Tomlin, C. and Degani, A. (2003). Discrete abstraction of hybrid systems: verification of safety and application to user-interfaces. NASA Technical Memorandum 2003-212803, Moffett Field, CA.
6. Heymann, M. and Degani, A., (2002). On abstractions and simplifications in the design of human-automation interfaces. NASA Technical Memorandum 2002-21397, Moffett Field, CA.