Challenges of Testing for Critical Interactive Systems

Valéria Lelli

Challenges of Testing for Critical Interactive Systems

Valéria Lelli*
INSA Rennes,
IRISA/INRIA, Triskell. Rennes, France
valeria.lelli_leitao_dantas@inria.fr

***Abstract*—Interactive systems cover all systems that represent a bridge to enable the user interaction over an interface. The advance of technologies such as ubiquitous computing brings new interaction designs. The new interaction styles have illustrated the shift from design graphical user interfaces to design human-computer interactions (HCI). However, several approaches still focus on testing GUI instead of testing HCI. The goal of this thesis is to investigate the use of new concepts to automate the interactive systems testing by addressing the issues of critical interactive systems.**

Keywords-automated testing;test case generation; interactive systems

I. INTRODUCTION

Over the last decade, the diversification of human-computer interactions (HCIs) and input devices used in multiple domains illustrates the shift initiated from designing graphical user interfaces (GUIs) to designing HCIs.

Novel HCI techniques, models, and architectures are bringing HCIs as first-class objects (i.e. post-WIMP systems) in systems where mice, keyboards, and widget-oriented GUIs (i.e. WIMP systems) are no more adapted. Yet, interactive system testing approaches focus on testing GUI rather than on testing HCI. Several challenges become more complex regarding the post-WIMP interactions, such as the automatic test cases generation from models and the automatic execution, etc. This concern also affects interactive critical systems having more and more advanced interactive features.

This PhD thesis, conducted in collaboration with energy industry partners, aims at investigating the use of modern software development concepts (such as model-driven engineering) to define new breaking and efficient methods to automate the interactive systems testing regarding the constraints of critical systems. More precisely, our research work aim at investigating the following research hypotheses:

- **H1: modeling interactive systems.** Which formalisms to use for modeling interactive systems having advanced interactive features?
- **H2: generating test cases.** How to automate the test cases generation for interactive systems having advanced interactive features?
- **H3: critical systems.** How can H1 and H2 be applied to critical systems?

II. CHALLENGES FOR INTERACTIVE SYSTEMS

Current approaches focus on test case generation for interactive systems looking at the GUI by manipulating only widgets, without considering new advanced features (post-WIMP) and specificities of critical systems. Several research papers have focused on automating GUI testing by adopting model-based testing (MBT) [7]. MBT has been applied to build models from specifications and generate GUI test cases to test them against the SUT (see Fig. 1). Thus, to provide a fully automated testing process for interactive systems several problems have to be faced as described below.

A. Building models representing HCIs

Models should be built to cover all specifications of the SUT, user interactions included, to enable an efficient automated test case generation. The main difficulty is to choose a suitable representation.

For example, GUI test models can be built by using capture/playback tools. However, a major limitation of this approach is that the models are build from the SUT itself. These test models can be represented by finite state machines (FSMs) or extended FSM (e.g., Markov chain), graphs, or formal languages. Several MBT approaches have adopted FSM to represent behavior models of the SUT. The main reason behind this adoption is the lack of expressiveness and abstraction in FSM structure since it can address many problems that cannot be solved in other languages [5].

Additionally, several approaches focus on using as entry points of the test process models describing the GUIs. Such GUI models are then transformed into GUI test model to ease the test case generation process. Nguyen et al. [1] propose an approach that builds models into two layers by separating the presentation from business logical and mapping these models automatically to generate the test cases. Some architectures for interactive systems (e.g., Malai [2]) enable to model these systems in different layers (e.g., presentation, HCIs) for several purposes such as documentation, communication, or code generation. For domains such as mobile computing where the applications support different HCIs, adopting this strategy is mandatory.

B. Generating automatically concrete test cases from models

Adopting an online approach instead of offline to build models has been used to achieve an effective test case generation. For example, Mariani et al. [3] propose a method to generate test cases online from a behavior model that is built incrementally while interacting with the GUI of the SUT.

Additionally, building models with a higher level of abstraction produces models that are not capable of covering all implementation details since some solutions depend on

*Co-advised by Benoit Baudry and Arnaud Blouin

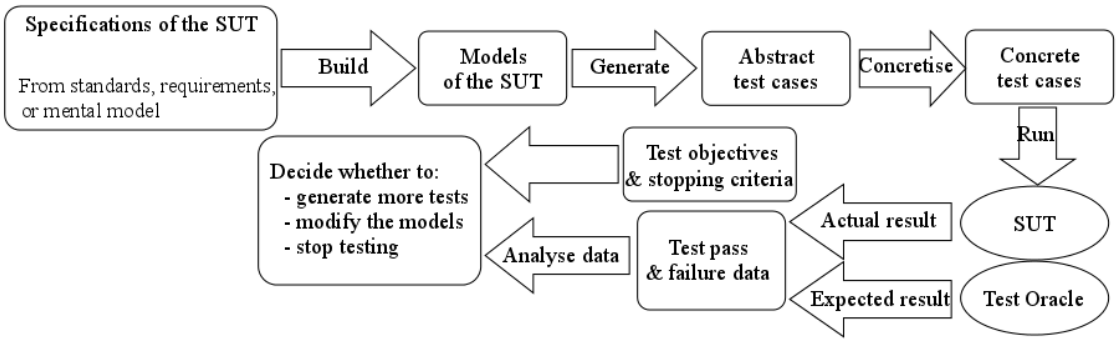

Figure 1 – Model-based testing process

the specific decisions from the target platform. To bridge this gap abstract test cases must be generated from the initial models [6]. Bowen and Reeves [4] present an approach to generate abstract tests from a formal model written in the Z language. However, abstract test cases do not specify concrete input data such as coordinates of a click. Thus, concrete test cases are generated (manually or automatically) from abstract test cases by defining concrete input data. Concretising automatically abstract test cases is a challenge for interactive systems since from one abstract test case can be generated several concrete test cases.

The test case explosion problem becomes more critical for interactive systems because of the diversification of HCIs. Modeling all possible HCIs of the SUT can increase the number of states exponentially in two ways: an extremely large state space or an infinite state space that is infeasible to analyse entirely. Thus, the test case generation for interactive system has to be managed to support the concrete test cases generation.

C. Testing critical interactive systems

During the development of critical interactive systems, such as command centers of power plants, the testing process is a critical step. One challenge is to explore how the previous challenges also concern and can be applied to critical interactive systems. Another challenge is the validation of the conformance against the requirements, mainly standards (e.g., IEC 61513), of the final system. For example, in these systems the operator interface contains an amount of information which is shown in different displays by using a range of technologies (e.g., LCD, CRT). This information must to be coherent in all displays to avoid failures provoked by the operator. On the other hand, to automate this testing process, a non-intrusive technique should be adopted to conform the security requirements dictated by those standards.

As we discussed above, current approaches to automate test cases generation for interactive systems do not cover advanced interactive features (i.e. post-WIMP) and critical issues. Thus, this thesis aims at addressing this problem.

III. EXPECTED CONTRIBUTIONS

Regarding the previously explained challenges, the possible contributions of the PhD thesis can be:

- Buiding models that can represent new advanced user interactions;
- From these models, generate automatically concrete test cases for post-wimp interactions;
- Propose an approach that addresses the issues related to critical systems domain.

We are using Malai architecture model to generate abstract test cases automatically for interactive systems at post-WIMP interactions level. Then, such abstract test cases must be concretised to be executed and address the issues for critical interactive systems. Our approach will be validated with critical interactions for instrumentation and control.

IV. CONCLUSIONS

This paper presents challenges for interactive systems in order to propose an approach to automate the test cases generation. Our intention is to apply our approach in the critical interactive systems domain by addressing issues intrinsic of these systems. The effectiveness of our approach will be evaluated in an industrial project.

REFERENCES

[1] D. H. Nguyen, P. Strooper, and J. G. Suess. 2010. Model-based testing of multiple GUI variants using the GUI test generator. *In Proc. of the 5th Workshop on* AST '10. ACM, NY, USA, 24-30.

[2] A. Blouin and O. Beaudoux. Improving modularity and usability of interactive systems with Malai. In Proc. of EICS'10, pages 115–124, 2010.

[3] L. Mariani, M. Pezzè, O. Riganelli, and M. Santoro. 2012. AutoBlackTest: Automatic Black-Box Testing of Interactive Applications. In Proc. of ICST '12, Montreal, Canada, 2012. IEEE Computer Society, pages 81-90.

[4] J. Bowen and S. Reeves. UI-driven test-first development of interactive systems. In Proc. of the EICS '11. ACM, NY, USA, pages 165-174, 2011.

[5] R. M. Hierons, K. Bogdanov, J. P. Bowen, R. Cleaveland, J. Derrick, J. Dick, M. Gheorghe, M. Harman, K. Kapoor, P. Krause, Gerald Lüttgen, A. J. H. Simons, S. Vilkomir, M. R. Woodward, and H. Zedan. 2009. Using formal specifications to support testing. *ACM Comput. Surv.* 41, 2, Article 9 (February 2009), 76 pages.

[6] J. L. Silva, J. C. Campos, and A. C. R. Paiva. Model-based user interface testing with Spec Explorer and ConcurTaskTrees. Electron. Notes Theor. Comput. Sci. 208. 77-93, 2008.

[7] A. C. Dias Neto, R. Subramanyan, M. Vieira, and G. H. Travassos. A survey on model-based testing approaches: a systematic review. In In Proc. of workshop WEASEL'07 , pages 31–36. ACM, 2007.