STER: A Strategy for TEsting Reactive Systems

Eliane Martins
UNICAMP
eliane@ic.unicamp.br

Daniele C. Guimarães
UNICAMP/INPE
daniele@dss.inpe.br

Ana Maria Ambrosio
INPE
ana@dss.inpe.br

Abstract
This article reports our current work in defining a specification-based testing strategy for reactive systems, named STER. The strategy is aimed at providing practitioners with a formal and systematic method of testing, which may be easily understood and applied in the industry. It is based on risk analysis to help pinpoint the critical parts of the system. Moreover, it guides testers to deal with limited testing time and resources common in industry context, without losing system quality. The experiments with the strategy has been supported by ATIFS [1], a test tool-kit of previous research, for testing some space applications developed at the National Institute for Space Research (INPE). Extensions in the strategy have been considered by aggregating fault injection-based test cases to evaluate the system also in the presence of faults.

Backgrounds
Reactive systems are often safety-critical and must respond continually to stimuli from their environment: computation and outputs are driven by inputs received from the environment [2]. In space applications, reactive systems are predominant to control and monitor on board equipment health, orbit and attitude positioning, ground station equipment monitoring and to autonomously operate satellites.
These systems are usually complex, concurrent, distributed and the number of potential input sequences that they must handle can be considered as infinite. In spite of these difficulties, reactive systems must be thoroughly validated to assure that it complies the required qualities. The cost of testing accounts for more than 50% of development costs. The activities of generating and selecting tests strongly contribute to this overall system testing cost. An approach for efficient test case generation, supported by tools, is thus mandatory to assure quality, specially for highly complex systems.
A set of test case generation methods have been developed for several formal models such as FSM, Petri Net, Statecharts, SDL, etc. However these methods are not commonly adopted by the industry yet. There are various reasons for that (c.f. [2, 3]). One of them is the difficulty in the use of formal methods for practitioners. Another is that most methods require a complete, consistent specification of the system, which is hard to obtain in practice. The huge amount of test cases that can be automatically generated is another practical problem.
The strategy we propose aims at coping with these problems. It is based on UML notations (which has becoming popular among developers); it is based on test case generation from a formal specification, but not for the entire system. The idea is to concentrate the test effort where it can most effectively reduce risk. According to the Pareto principle, 80% of all errors uncovered during testing will likely be traceable to 20% of all program modules [4]. Adopting a risk-based analysis before generating the system test cases, one may find the risky parts over the whole system and then to thoroughly test these parts.
In the following we present some of the problems addressed by STER, the approaches that inspired STER, its steps and finally the ongoing work.

Addressed problems
When systematically testing a reactive system based on its specification, in practice one has to address a lot of issues [2, 3]. Some of them, addressed by STER, are described in the following.
One of the first issues is the use of *formal methods*. Some formalisms being defined to specify a reactive system are based on sound mathematical theories that allow to formally verify whether the specification has some desired properties. However these formalisms are hardly applicable to real complex systems. Also, formal methods are still difficult to put into practice. To cope with this issue, STER is based on various artifacts generally produced during the Analysis and Design phases, thus allowing the re-use of many already existing models. Since UML

notations are mostly used in practice, these are the ones considered in our strategy.

Another difficulty with formal testing methods is that most of them assume a relatively stable and *complete specification*. This is often impossible in practice. In many specifications only some functions or aspects are (or can be) formally specified. Besides, requirements change over time: the desired end system is rarely clearly and completely defined. STER allows users to partition the system according to use cases, so tests can be considered incrementally, firstly only the use cases completely specified are tested.

Another problem that makes specification-based testing unfeasible in practice is the *state explosion* problem. This can occur: (i) due to the parameters of the interactions of a system, which implies that the system behavior depends on constraints on these values, (ii) due to concurrency - the behavior of the whole system is given by the composition of the behavior of its components. To cope up with this problem STER proposes a risk-based testing approach: only the use cases considered more critical are further detailed and thoroughly tested. Test effort is thus concentrated on those parts of the system whose failures can heavily impact system behavior.

Existing test generation algorithms can generate a *huge number of test cases*. A selection is thus mandatory because of time and resource constraints. The risk analysis also applies to scenarios of a use case: test cases are generated for the most critical scenarios of each use case considered for testing. In addition, STER is an incremental testing strategy: first, each critical use case is tested in isolation. Then they are integrated with other use cases for the testing the whole system.

Related work

Some system testing approaches have inspired the strategy adopted in STER: TOTEM [5] - based on various UML artifacts, from which we adopted the use case sequence for testing the whole system; SCENT [6] - based on scenarios, with which a state model is obtained; ETACS [7] - use risk analysis; and the strategy in [8] – define a test strategy based on the user profile.

The STER strategy

The STER steps are as follows:

1. Generate the use cases describing the system.
2. Define the risk (weight) of each use case.
3. Derive scenarios from the most risky use cases.
4. Define the risk of each scenario for the selected use cases.
5. Derive a sequence diagram to represent the scenarios.
6. Derive a finite state machine (FSM) representing the most risky scenarios for each selected use case.
7. Generate a test sequence from the FSM, using a tool developed for that purpose. The test cases will be used to test each selected (critical) use case in isolation.
8. Derive test cases for the system, by combining the test cases obtained for the individual critical use cases. Sequences are derived from Activity Diagrams representing the flow of use case executions for each actor, as in [5].

Ongoing works

Preliminary results in an artificial case study has shown the usefulness of the strategy. Moreover, it is being used for testing a satellite operation system developed at INPE. For space application compliance, STER is being extended with guides for fault-injection based test cases definition. The fault injection purposes are: (i) to test specified exception outputs as another means to avoid state space explosion; and (ii) to test the system behavior under unspecified situations.

References

[1] in site: http:www.inpe.br/atifs

[2] L.J. Jagadeesan, A. Porter, C. Puchol, J. C. Ramming, L. G. Votta, "Specification-based Testing of Reactive Software: A Case Study in Technology Transfer". Proc. of the 19th International Conference on SE, May,1997.

[3] R. Lai, W. Leung, "Industrial and academic protocol testing: the gap and the means of convergence". Computer Networks and ISDN Systems, 27, pp537-547, 1995.

[4] Pressman, R. S., "Software Engineering – A practitioner's Approach". McGraw-Hill, 5ª ed., 2001.

[5] L. Briand, Y. Labiche, "A UML-Based Approach to System Testing", Technical Report SCE-01-01, Version 4, Carleton University, 2002.

[6] J. Ryser, M. Glinz, "SCENT: A Method Employing Scenarios to Systematically Derive Test Cases for System Test". Technical Report 2000/03, Institut für Informatik, Universität Zürich.

[7] L. M. Volpi, "Uma estratégia de teste de software para Ambiente Cliente-Servidor". Dissertação de Mestrado.Universidade Federal do Paraná, 2001.

[8] J. McGregor, M. Major, "Selecting Test Cases Based on User Priorities", 2000.