Testing interruptions in reactive systems

Wilkerson L. Andrade and Patrícia D. L. Machado

Software Practices Laboratory (SPLAB), Systems and Computing Department (DSC), Federal University of Campina Grande (UFCG), Av. Aprígio Veloso, 882 Bairro Universitário, CEP 58429-900, Campina Grande, PB, Brazil. E-mail: wilkerson@computacao.ufcg.edu.br

Abstract. Modelling and testing of reactive systems with interruptions are discussed. These systems are commonly found in portable devices, where interruptions to a running application can be demanded at any time, due to concurrent execution of processes sharing a single resource, such as screen, as well as arrival of calls from network distributed services. Since the possible number of combinations of allowed interruptions is large, proper test case selection activities need to be performed. But, in order to systematically investigate and select test cases, it is fundamental to explicitly model interruption behaviour in a compositional way, avoiding the need for explicit enumeration. This work presents a strategy for testing interruptions in reactive systems that covers modelling for testing of systems with interruptions, generation and selection of sound test cases. The strategy is supported by the LTS-BT tool. Moreover, a formal model of an environment devoted to execution of test cases with interruptions is presented. Finally, a case study illustrates its applicability in the mobile phone application domain.

Keywords: Model-based testing, Test case generation, Interruption, Reactive systems

1. Introduction

With the growth and widespread use of portable devices such as mobile phones and handhelds, reactive systems that interact with their environment by accepting inputs and producing outputs have become more and more complex. The reason is that these devices usually operate with limited resources, accepting network services and running concurrent tasks. In this context, interruptions are usually applied so that services can be activated as soon as demanded. For this, the task running in the foreground is instantly suspended to release resources for the interrupting task. After interruption, the interrupted task should resume from the point where it stopped [AILS07, Hal93, Lap04, LY03]. As an example, when a user is composing an e-mail by using a mobile phone device and an incoming call arrives in this device, the call feature interrupts the e-mail feature that must successfully resume later.
Considering that any interruption can occur at any point of a flow of execution, there are infinite possibilities of occurrences. This makes the exhaustive specification of each possibility infeasible and, consequently, automatic test case generation and selection is compromised. Effective testing requires a systematic investigation of all possibilities. This cannot be successfully performed unless automation is possible.

To provide an effective solution for interruption testing, it is crucial to define a model capable of representing such interruptions, and consequently, make the automatic test case generation process possible. In addition, the model has to be composable, allowing interruptions to be combined at different points of possibly different flows of execution. Moreover, due to the large amount of possible test cases, selection strategies need to be applied to reduce the size of test suites. Furthermore, the test execution environment should be carefully considered so that execution requirements and constraints are properly identified and handled.

The particular problem of evaluating if a system implementation is in accordance with its specification by experimentation is referred to as conformance testing. Regarding reactive systems, considerable progress has already been made in this area from both theoretical and practical point of view. The AGEDIS project [HN04] is an outstanding initiative. Nevertheless, to the best of our knowledge, approaches that handle systems with interruptions are practically nonexistent.

This paper presents a strategy for conformance testing of reactive systems with interruptions that covers modelling (devoted to testing), generation and selection of sound test cases. The model adopted is named Annotated Labelled Transition System (ALTS). This kind of Labelled Transition System (LTS) has special descriptions inserted into the model in order to make the test case generation process feasible. LTSs are good models for functional testing because all information needed is the observable interactions between applications and environment and between applications. Also, they are the underlying formalism of most formal notations for reactive applications. The proposed model is implemented by the LTS-BT tool [CANM08]. A case study illustrates the benefits of the strategy when compared to manual selection. This paper extends the work presented in [AM09] according to the following aspects: (1) algorithms defined to translate high level specifications into ALTS models are presented; (2) a CSP formal model of an environment devoted to execution of test cases with interruptions introduced in [dFAM06] is instanced in the context of ALTS test cases; (3) the case study is presented in more details and results are more thoroughly discussed.

The remainder of this paper is structured as follows. Section 2 presents the general test process considered. Section 3 presents the ALTS behavioural model structure used to model interruptions and the test execution model. The interruption test case generation algorithms and a selection strategy based on test purposes are introduced in Sect. 4. In Sect. 5, some properties of interruption test cases generated by LTS-BT are discussed. A case study is presented in Sect. 6. Finally, Sect. 7 presents related work and Sect. 8 concluding remarks.

2. Context

In general, the test process in the context of this work starts with the specification of the System Under Test (SUT) and interruptions. Given high level specifications, an ALTS model is automatically generated. Finally, the ALTS model is combined with test purposes for interruption test case generation. The interruption test process uses test purposes in order to test at specific points of interest. A general view of this test process is presented in Fig. 1. This process considers the test architecture presented in Fig. 2. In this test architecture, two elements are important: the SUT and the TESTER. The SUT is composed of the main application and interruptions allowed during the test process. The environment is assumed to be fully controllable by the TESTER, thus, during test execution the TESTER controls all the interruptions, deciding when they start and finish.

The SUT is specified as use cases using a controlled natural language [CS08, LTB07, TLB06]. An example of a use case of a mobile phone application is shown in Fig. 3. This represents the behaviour of removing a message from inbox. A use case must have a main flow and can have some alternative flows. The flows are described through steps that include a user action and the respective system response. For instance, the step “4M” has the selection of the “Remove” option, and the respective system response is to show an alert saying that the message was removed.

Besides the actor action and the system response, each step has a condition (System State) that determines if the system response will happen or not. If the condition is not satisfied, an alternative flow must be specified. As an example, the step “4M” of the main flow has one alternative flow (steps “1A” and “2A”).