Enabling statistical testing for component-based systems

Thomas Bauer, Robert Eschbach

Fraunhofer-Institut für Experimentelles Software Engineering IESE
Fraunhofer Platz 1
67663 Kaiserslautern, Germany
{thomas.bauer | robert.eschbach}@iese.fraunhofer.de

Abstract: In this article, an automated statistical test approach for the systematic quality assurance of component-based systems is presented. From state-based component models a compact interaction test model is constructed, which describes component interactions and interaction sequences. The model is annotated with an interaction profile to facilitate the generation of representative test cases and the reliability estimation of the test object during the test evaluation. For the test case generation and reliability estimation model-based statistical testing was applied. A small example of a component-based system from the automotive domain is given.

1 Introduction

In component-based systems, faulty component interactions have a high impact on the dependability of the system. With the component-based development of software systems and the use of third-party components, the integration and interoperability testing of a distributed system has become an important quality assurance activity. Integration testing is performed in parallel to the integration stage where the system is composed from its components. The goal is to test system interactions and functionalities that are spread across several system components. Our work is restricted to interactions that are provoked by the event flow between components and the synchronization of components on certain events. Furthermore, we assume that state-based component models are available.

We propose an integration test approach that fully covers the component interactions and sequences of interactions by constructing a compact *interaction test model.* The model is derived from state-based component models that represent the functional component requirements. The interaction test model is enhanced by an *interaction profile* to enable statistical testing, generate representative sets of test cases, and estimate the product reliability based on the test results. The reliability estimates can be used as test stopping criterion, to assess the current status of the product quality, and to predict future problems after the product release.

The contribution of this article are the construction approach of interaction test models from state-based component models and the application of statistical testing to test component-based systems and assess the reliability w.r.t. component interactions. This paper is structured as follows: In chapter 2, related work in the field of model-based integration testing is presented. Our general approach is described in chapter 3. Chapter 4 contains a detailed description of the construction procedure of the interaction test models. The application of statistical testing and the usage profiles are covered by chapter 5. The article concludes with a summary and outlook in chapter 6

2 Related Work

The research in integration testing comprises a wide field of systematic test solutions for component-based systems. Interaction profiles and reliability estimations have not been covered. Most of the automated test approaches use different types of models, e.g. requirements models, component behavior models, system structure models, or code models. Several approaches have been developed which generate test cases from state-based component models [ABJ07, BRT03, DV07, HIM00, HP05, LR01]. These approaches focus on the efficient coverage of single component interactions. Longer sequences of component interactions are not considered. Some publications address the complexity in state machine composition during the integration testing. The approach in [RHP08] creates partial models from component state machines. Model checking is used to derive test cases based on several communication coverage criteria. The approach also works for extended finite state machine (EFSMs) with variables and conditions. [Pre03] proposes a solution for the generation of integration test cases from component test cases. He uses EFSMs for describing the component functionalities. The resulting integration test suite provides MC/DC (Modified Condition/Decision Coverage) of the underlying component models. [KCT02] aimed at reducing the number of states and transitions of composed state machines by incrementally creating reduced observationally equivalent state machines. The models were derived from code models (control and data flow models).

Some integration test approaches combine high level system models with low level component models. [SWK09] uses a so-called global choreography model for system modeling and local partner models for component modeling. Abstract test cases are generated from the global model and later refined into traces of the component models by applying model checking. [Be07] uses feature interaction models for the high level description and low level component models. Test cases are generated based on several feature interaction coverage criteria. Both solutions avoid the construction of the complete product model, but they do not support the systematic generation of longer interaction sequences. The generation of representative test cases by incorporating a usage profile is also not considered. .

3 Approach Overview

The main idea of our approach is the development of an interaction test model from component models. The interaction test model is a simplified system model which incorporates an interaction profile to describe importance and criticality of component interactions. Component test models (CTM) describe the possible input-output trajectories of components and theircontrol logic represented by a finite state machine. These models might be re-used from component testing or component design.

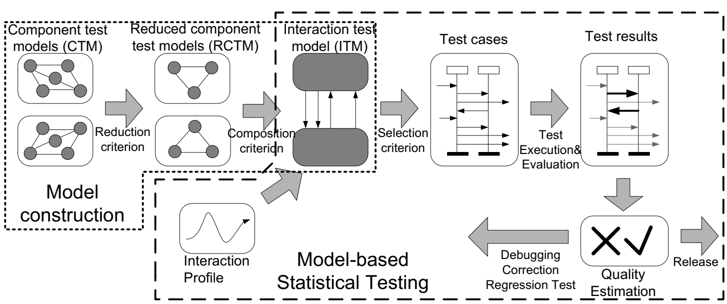

Figure 1: Steps of the test approach

Figure 1 shows the steps of our approach which is divided into two phases, model construction and model-based statistical testing. At first, CTMs are analyzed. All elements of the CTMs that are not needed for executing interactions and interaction sequences are removed. The remaining transitions are merged to simplify the models and to ease the model composition later. The resulting reduced component test model (RCTM) contains the condensed information that is required to generate valid sequences of component interactions from the component's perspective. The RCTM transitions are composite. Each of them contains a synchronized event, either as a stimulus or as a response. In the next step, the RCTMs are composed into the interaction test model (ITM) by applying product composition. Since all RCTM transitions contain synchronized events, the transitions are either merged or removed for the ITM. The ITM contains all reachable component interactions based on the underlying component test models. Due to the high number of component synchronizations, the resulting ITM becomes small compared to the complete product model which would be generated from the CTMs.

Additionally, the ITM is annotated with an interaction profile which describes the frequency and importance of different interactions. This allows the application of model-based statistical testing (MBST). The test case generation, execution, and evaluation is fully automated by the approach. Chapter 5 describes the application and adaptation of MBST in detail.

4 Model Construction

In order to systematically test component interactions and sequences of them, the possible connections of different component interactions are interesting. Looking at the CTMs we have identified three relevant aspects: the ways from the start state to the interacting transitions (*initial interaction paths*), the ways between two interacting transitions (*connecting interaction paths*), and the ways from an interacting transition to the exit state (*final interaction paths*). Each aspect can be described by a set of particular paths. These paths do not contain any synchronized transitions except for the last transition. This assures that other interactions are not triggered by accident during the test execution.

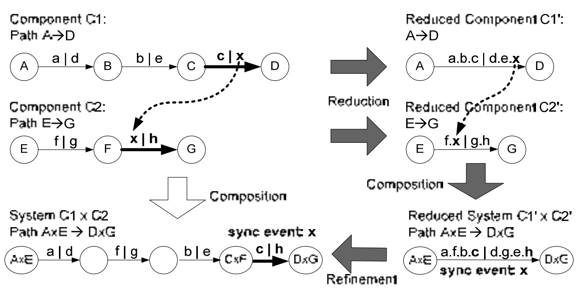

Figure 2: Model construction

In the first step, we extract the component-specific information that is needed to construct these three artifacts. Therefore, we have to determine all synchronized transitions of the CTMs, i.e., all transitions with synchronized events. Then, we determine the initial and final paths for each synchronized transition and the connecting paths for each pair of synchronized transitions. Based on that, the RCTMs can be generated containing the interaction-relevant information of the component. The RCTMs are composed into a so-called interaction test models (ITM). Figure 2 shows the mapping of CTM paths to transitions of the RCTM and their composition into transitions of the ITM. Assuming that A and E are the start states of the CTMs *C1* and *C2*, the paths in the figure represent the initial paths of *C1* and *C2*. Both models synchronize on event *x* when *C1* is in state *C* and *C2* in state *F*. In order to simplify the model structure, we combine the transition paths of the CTMs into single composite transitions of the RCTMs. Each transition of a RCTM represents either an initial path α, a connecting path β, or a final path γ. The composite transitions of the α- and β-paths contain exactly one component interaction at the end, in our example the flow of event *x*. The resulting RCTM will only contain the source and target states of the composite transitions. The other states can be easily determined by deriving the sequence of atomic transitions from the composite transition.

When composing two RCTMs, the last atomic transitions of the composite transitions may synchronize, here: $c|x$ of *C1* and $x|h$ of *C2*. The result in the product is an atomic transition $c|h$ with the internal event *x*. The non-synchronized atomic transitions of the underlying RCTMs are combined in series with preserved order. Here, we chose an interleaving approach for the *C1'* and *C2'* transitions, resulting in the input sequence *a.f.b* with the output *d.g.e*. The transition of the composed ITM *C1'xC2'* is also composite. It can be refined into a sequence of atomic transitions, which corresponds to a valid path in the unreduced product model *C1xC2*.

5 Model-based Statistical Testing

Model-based statistical testing (MBST) is a black-box testing technique that enables the generation of representative test cases from the tester's or user's perspective [WPT95]. The internal model is a discrete time Markov chain which describes the stimulation of the test object by its environment. The operational profile (in MBST usage profile) of the test object, which is annotated in the test model, represents frequency, criticality, or cost of the test execution [Mu93]. Based on the test results and the underlying operational profile the system reliability is estimated. For our ITM we wanted to define an appropriate *interaction profile* to support statistical testing of component interactions. The resulting interaction profile describes the probabilities of executing component interactions at different states of the ITM.

The example of the feasibility study, an in-car info- and entertainment system consisting of a phone and a CD-player, is shown is Figure 3. The system has to fulfill comfort functions which are provided by the interplay of both components. The interesting parts of the example deal with interrupting and resuming the CD-player during a phone call. For the first version we made the following simple assumptions: Incoming calls (60%) are more frequent than outgoing calls (40%). The CD-player is more often in the playing state (80%) than in the state when it is paused by a radio traffic message (20%). An incoming or outgoing phone call is rejected in 20% and accepted in 80% of the cases. The probabilities of the transitions to the exit state have a high impact on the average length of randomly generated test cases. The probability of the exit transition (γ-path, *stop+deact*) was set to 10% which leads to an average length of randomly test cases of about 21 events in the statistical model analysis. If longer or shorter test cases are required the probabilities of the exit transitions have to be adapted. More information on statistical model analysis and simulation is provided in [WPT95].

Test cases are automatically generated from the test models as arbitrary paths from the start to the exit state. Different strategies for automated test case generation exist, including model coverage for covering all states, transitions, or transition pairs and random tests which randomly traverse the model based on the usage profile. The number of test cases to be generated is linked to the planned test effort. Assuming an average test case length of 21 events (taken from the statistical model analysis), 10 sec for setting up a test case, 5 sec for executing and evaluating one composite transition of the ITM, the average time needed for preparing, executing, and evaluating a random test case is about 120 sec. If we consider a 2 hours slot for the execution of the test case set, about 60 test cases can be executed. In order to assure the complete coverage of all model states and transitions, we generate the transition coverage set first, followed by 60 random test cases.

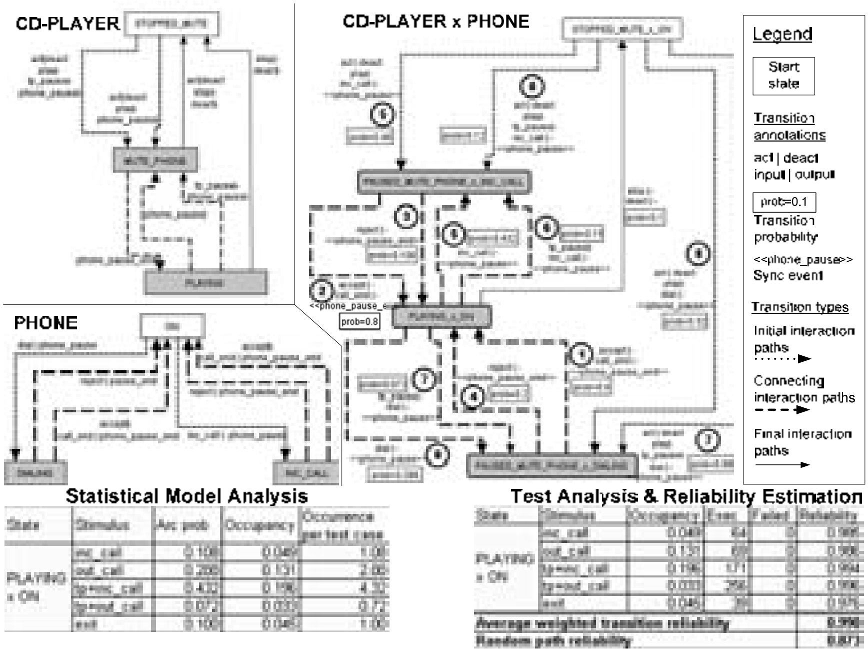

Figure 3: Example for component models and interaction test models

After the execution and evaluation of the test cases, the reliability of test object is estimated based on the test results [SP00, PP05]. The reliability estimation represents the probability that a future system usage will not fail. It represents the confidence of the user in the test object. During the test planning the test manager may set a desired reliability value as test stopping criterion, e.g. 0.99. Based on the interaction profile some composite transitions of the ITM are executed very often in the test, e.g. *[PAUSE_MUTE_PHONExINC_CALL].”accept+call_end”* with 289 occurrences in all test cases. The reliability estimates for this transition is very high assuming zero failures (0.997). Other transitions were generated and executed only few times, e.g. *[STOPPED_MUTExON]."act+play+tp_pause+dial"* with only 7 occurrences which leads to a reliability estimate of 0.889. The statistics from the reliability estimations are used to support the test management decisions, e.g. the effort distribution of regression tests. The overall system reliability is represented by a weighted average of the transition reliabilities. In our example the estimate is 0.990 which satisfies the test stopping criterion.

6 Conclusion and Outlook

We have presented a new approach for the systematic generation of interaction test cases from state-based component test models. In the first step, reduced component test models were constructed, focusing on the component interactions and the connections between different interactions. In the next step, the reduced models were composed into interaction test models. An interaction profile was annotated to enable the statistical generation of representative test cases. Finally, we applied structural coverage criteria and random search to automatically derive test cases. Based on the test results and the underlying interaction profile, the reliability w.r.t. component interactions could be estimated by applying model-based statistical testing.

The approach was initially applied to a simplified demonstrator from the automotive domain. In the feasibility study, we were able to significantly reduce the complexity of the models used for automated test case generation. The interaction profile allowed the generation representative test cases for the integration testing of a component-based system. The quality of automatically generated test cases especially the failure detection capabilities have to be assessed in a more complex industrial case study.

Future work will comprise the determination of an appropriate integration strategy based on the component models. Furthermore, we aim at improving the algorithms used for the construction of the interaction test models. In the current status of the work we have not considered the reliability of the underlying communication channels. This could be another research topic for the future.

This work was funded by the German Federal Ministry of Education and Research (BMBF) in the context of the project ViERforES (No.: 01 IM08003 B).

7 References

[ABJ07] Ali S., Briand L. C., Jaffar-ur Rehman M., Asghar H., Zafar Z., Nadeem A.: A state-based approach to integration testing based on UML models, Journal of Information and Software Technology (Elsevier), vol. 49(11), 2007

[Be07] Benz S., Combining test case generation for component and integration testing, 3rd International Workshop on Advances in Model-Based Testing 2007, London, pp.23-33, 2007

[BRT03] van der Bijl M., Rensink A., Tretmans J.: Compositional Testing with ioco. 3rd FATES, pp. 86-100, 2003

[DV07] Desmoulin A., Viho C.: A New Method for Interoperability Test Generation. 19th TestCom/FATES, pp.58-73, 2007

[HIM00] Hartmann, J.: Imoberdorf, C.; Meisinger, M.: UML-Based Integration Testing, In: Proc. of the Intl. Symposium on Software Testing and Analysis Portland, USA, pp.60-70, ACM Press, 2000.

[HP05] Huo j., Petrenko A.: Covering Transitions of Concurrent Systems through Queues. 16th ISSRE, pp. 335-345, 2005

[KCT02] Koppol P. V., Carver R. H., Tai K.-C.: Incremental integration testing of concurrent programs. IEEE Transactions on Software Engineering, 28(6):607–623, 2002

[LR01] Lyer S., Ramesh S.: Apportioning: a technique for efficient reachability analysis of concurrent object-oriented programs. IEEE Transactions on Software Engineering, 27(1):1037–1056, 2001.

[Mu93] Musa, J.D., “Operational Profiles in Software Reliability Engineering,” IEEE Software Magazine, vol 10(2), p.14-32, 1993

[PP05] S. Prowell, J. Poore, Computing system reliability using Markov chain usage models, Journal of Systems and Software, Vol. 73, No. 2, October 2004, 215 - 225.

[RHP08] Robinson-Mallett C., Hierons R., Poore J., Liggesmeyer P.: Using communication coverage criteria and partial model generation to assist software integration testing. Software Quality Journal 16(2): 185-211 (2008)

[Pr03] Pretschner A.: Compositional Generation of MC/DC Integration Test Suites. Electr. Notes Theor. Comput. Sci. 82(6): (2003)

[SP00] K. Sayre, J. Poore, "Stopping criteria for statistical testing", Information and Software Technology, Vol. 42, No. 12, September 2000

[SWK09] Stefanescu A., Wieczorek S.,Kirshin A.. MBT4Chor: A model-based testing approach for service choreographies. ECMDA-FA 2009, vol.5562, Springer LNCS, pp. 313-324, 2009

[WPT95] Walton G., Poore J., Trammell C.: Statistical Testing of Software Based on a Usage Model. Softw., Pract. Exper. 25(1): 97-108 (1995)