**Heidelberg University**
**Institute of Computer Science**
**Software Engineering Group**

**Better Software Through Systematic Testing**

# Literature Synthesis

`https://github.com/fidsusj/SWE-Seminar`

Felix Hausberger, 3661293,
Applied Computer Science
eb260@stud.uni-heidelberg.de

# 1 Synthesis

## 1.1 Summary

**Automatic Test Generation: A Use Case Driven Approach**. The paper addresses the traceability problems between high-level use cases and concrete test case execution as well as the lack of integration of formal specification methods with well-established development life cycles. It shows a method to automatically generate executable test case scenarios by making use of a transition system and therefore shifting the effort of test generation to the specification activity. In the first step, UML use cases get enhanced with contracts (pre- and postconditions). The contracts are made executable by writing them in the form of requirement-level logical expressions. Through exhaustive simulation a transition system is built, which serves as a model for all valid sequences of use cases. Relevant test objectives get extracted from the transition system by applying predefined coverage criteria. Subsequently in the second step, test scenarios get generated by replacing each use case in a test objective with a use case scenario that is compatible in terms of static contract matching. The process results in executable test scenarios that get evaluated using the statement coverage metric.

**An Automated Approach to System Testing based on Scenarios and Operations Contracts**. Based on the suggested improvements in the first paper, the second paper uses interaction overview diagrams (IOD), a special form of activity diagram used to show control flow, to derive test paths. It helps to start testing in early stages of development. The IODs get enhanced with contracts written in the object constraint language (OCL) and the get transformed into a contracts transition system (CTS) which models all scenarios of the IOD. Through traversing the CTS test paths get derived. Key difference to the first paper is that test scenarios do not get generated on system level but rather on use case level due to the fact that contracts are not attached to use cases but to IODs. It therefore serves as a platform to generate more in-depth test scenarios (as well for negative test cases).

## 1.2 Synthesis matrix

1. Description of the approach (What does the approach do?)
    (a) Which artifacts and relations between artifacts are used in this approach? Which artifacts are created in the course of the approach? How are the artifacts characterized?
    (b) What is required and/or input for the application of the approach?
    (c) What steps does the approach consist of? Which information is used in which step and how? What are the results of the individual steps?
2. Benefits of the approach (Who does the approach help and how?)
    (a) Which usage scenarios are supported by the approach?
    (b) Which stakeholders are supported by the usage scenarios?
    (c) Which knowledge areas from SWEBOK can be assigned to the usage scenarios?
3. Tool support for the approach (What tool support is available?)
    (a) What tool support is provided for the approach?
    (b) Which steps of the approach are automated by a tool? Which steps are supported by a tool, but still have to be executed manually? Which steps are not supported by a tool?
4. Quality of the approach (How well does the approach work?)
    (a) How was the approach evaluated?
    (b) What are the (main) results of the evaluation?

| Nr. | Approach [1] | Approach [2] |
|---|---|---|
| 3a | Use cases describing the basic operations in the transition system. Contracts that are attached to the use cases describing the states in the transition system. The transition system (UCTS) itself as a simulation model to derive test objectives from. Test objectives describing the test paths. UC-System as a third party tool to build the UCTS and to derive test objectives. Use case scenarios to build test scenarios from test objectives. UC-SCSystem to generate executable test scenarios. | Use cases describing the basic operations in the transition system. Contracts that are attached to the use cases describing the states in the transition system. The transition system (UCTS) itself as a simulation model to derive test objectives from. Test objectives describing the test paths. UC-System as a third party tool to build the UCTS and to derive test objectives. Use case scenarios to build test scenarios from test objectives. UC-SCSystem to generate executable test scenarios. |
| 3b | tipps | tricks |
| 3c | tipps | tricks |
| 4a | tipps | tricks |
| 4b | tipps | tricks |
| 4c | tipps | tricks |
| 5a | tipps | tricks |
| 5b | tipps | tricks |
| 6a | tipps | tricks |
| 6b | tipps | tricks |

## References

[1] Clémentine Nebut, Franck Fleurey, Yves Le Traon, Jean-Marc Jézéquel, “Automatic test generation: A use case driven approach,” vol. 32, 2006. [Online]. Available: `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1610607` (visited on 12/06/2020) (cit. on p. 3).

[2] Najla Raza, Aamer Nadeem, Muhammad Zohaib Z. Iqbal, Ed., *An Automated Approach to System Testing based on Scenarios and Operations Contracts*, IEEE, 2007. [Online]. Available: `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4385504` (visited on 12/06/2020) (cit. on p. 3).