EXECUTION OF UTP TEST CASES USING FUML

Niels Hoppe
EXE 2018 in Copenhagen, Denmark
October 14th, 2018
Niels Hoppe
Student in working group of Marc-Florian Wendland, SQC
niels.hoppe@fokus.fraunhofer.de
Co-author
INTRODUCTION

Agenda

1. Introduction and motivation
2. Executable UTP test models
3. Mapping and transformation
4. Execution in Moka
5. Conclusion
Influencing standards

INTRODUCTION TO UTP

Conceptual standards
ISO 29119
ISTQB syllabi
ETSI MBT

Technical standards
ETSI TTCN-3
UML
UTP 1.0

Support for domain-specific standards
Do-178C  ISO/IEC 61508  EN 50129  ISO 26262  …
Conceptual overview of UTP

Test Objective | Test Requirement | Test Context
--- | --- | ---

Test Management / Test Planning (dynamic test process)

UTP 2

Test Generation Layer
- Test Design Input
- Test Design Directive
- Test Design Technique
- Test Data Specification

Test Definition Layer
- Test Configuration
- Test Case
- Test Set
- Test Data Values
- Test Item
- Test Component

Test Execution Layer
- Test Execution Schedule
- Test Actions
- Test Log Structure
- Test Log
- Arbitration Specification

INTRODUCTION TO UTP

© Fraunhofer FOKUS

Fraunhofer FOKUS
MOTIVATION

Classic Software Engineering

- Requirements
  - System
  - Test System
Model-Driven Software Engineering

- Requirements
  - System model
    - System
    - UML, SysML, etc.
  - Test model
    - Test System
    - TTCN-3, JUnit, Robot, etc.
  - UML Testing Profile
MOTIVATION

Model-Based Shift Left Testing

UML, SysML, fUML, etc.
→ Executable specification

System model

Requirements

Test model

System

Test System

UML Testing Profile
→ Execution semantics based on fUML and PSCS
Contributions of this paper

1. Requirements and constraints for executable UTP test models
2. Examples for executable UTP test model and adaptation model
3. Mapping and QVTo transformation
4. Execution environment based on Eclipse and Moka
EXECUTABLE UTP TEST MODELS

Structure overview

➢ Test context
  ➢ Test set 1
    ➢ Test configuration
    ➢ Test case 1.1
    ➢ Test case 1.2
  ➢ Test set 2
Test configuration

EXECUTABLE UTP TEST MODELS
EXECUTABLE UTP TEST MODELS

Test configuration and test cases

- «Component»
  - «TestConfiguration»
    - Environment

- «TestItem»
  + elevator: Elevator [1]

- «TestComponent»
  + tester1: Tester1 [1]

- «TestComponent»
  + tester2: Tester2 [1]

- «TestCase»
  sd: Test case

- «CreateStimulusAction»
  CallSignal(floor: Integer = 0, direction: Direction = UP)
EXECUTABLE UTP TEST MODELS

Test case behavior

«TestCase»
sd: Test case

tester1-Tester1  «CreateStimulusAction»
CallSignal(floor: Integer = 0, direction: Direction = UP)

tester2-Tester2

«ExpectResponseAction»
ResponseSignal(foo: Boolean = true, bar: Boolean = false)

elevator-Elevator

«CreateStimulusAction»
OpSignal(floor: Integer = 1)

«ExpectResponseAction»
ResponseSignal(foo: Boolean = true, bar: Boolean = false)
Transformation

- Transformation from **platform-independent** test model to **platform-specific** test model
- Platform is fUML and PSCS
- Mapping rules for:
  - Test sets
  - Test cases
  - Test components and actions
Test sets

• (Structure)

• Behaviors:
  1. Setup
  2. Teardown
  3. Main
  4. Factory
Test cases

• Structure
  1. Resolution of Generalization
  2. Property coordinator: Component to coordinate test components
  3. Owned Connector: Synchronization Bus

• Behaviors:
  1. Setup
  2. Teardown
  3. Main
  4. Factory
  5. <<Create>> Constructor
Test cases
Test cases
Test cases
Test cases

TC Factory

Create TC_1

result

Start TC_1

object

Accept

TC_1CompletionSignalEvent

End
**Test case coordinator**

- **Structure**
  1. Property testCase: Reference to containing test case

- **Behavior**
  1. Main
     - Synchronization of test components through StartSignal and CompletionSignal
     - Distribute test case log to test components
Test components

• (Structure)
• Behaviors:
  1. Main, implements test actions
     • CreateStimulus
     • ExpectResponse
     • Logging
     • Synchronization via GeneralOrderings
**Test components**

- (Structure)
- Behaviors:
  1. Main, implements test actions
    - **CreateStimulus**
    - ExpectResponse
    - **Logging**
    - Synchronization via GeneralOrderings
Test components

- (Structure)
- Behaviors
  1. Main, implements test actions
    - CreateStimulus
    - ExpectResponse
    - Logging
    - Synchronization via GeneralOrderings
EXECUTION IN MOKA

Extension and customization of Moka

• Implementation of OpaqueBehaviors in Java:
  • Initialization and finalization of test logs
  • Creation of test log entries
• Construction strategy for N-ary Connectors
EXECUTION IN MOKA

Logs generated during execution

- Model Explorer
- Log Model
  - TestSetLogs for Two testers with adapter
    - TestCaseLogs for TC_1
      - «TestCaseLog» TCL @ 1522156580953
        - <Slot> floor: 0
        - <Slot> direction: UP
      - «CreateStimulusLogEntry» MELE @ 1522156608336
        - <Slot> floor: 1

- «TestCaseLog» TCL @ 1522156580953
  - Applied stereotypes:
    - executingEntity: ValueSpecification [*] = []
    - executionDuration: Duration [0..1] = null
    - testLogEntry: TestLogEntry [*] = [MELE @ 1522156608336,
      predict: ValueSpecification [0..1] = null]
Achievements

- First definition of executable subset of UTP
- Mapping and transformation
- Execution and logging
- First steps towards precise execution semantics for UML Testing Profile

Next steps

- Integration with analysis and arbitration of logs
- Support for operations, complex actions, etc.
- Proof of concept for real-world adapter
- Improve efficiency of implementation