

Test Automation as a Model-Driven Engineering Process with MDTester

“Test Automation is Model-Driven Software Development”

Alain-G. Vouffo Feudjio

FOKUS (Fraunhofer Institute for Open Communication Systems)

Abstract. Recent progress in model-driven engineering (MDE) have made the perspective of applying the same type of approach for test automation yet more attractive. The model-driven testing (MDT) approach consists of designing platform-independent test (PIT) models at a high level of abstraction and transforming those automatically through platform-specific test (PST) models into executable test scripts. However, intelligent tool-support is a critical pre-condition to making that vision true. Beyond the fact that the process of modelling for test automation must be intuitive and efficient, it must also be ensured that the created test models are readable, valid and self-consistent. Furthermore, automated transformation of those models into executable test scripts and backwards are required to ensure round-trip engineering and a reuse of legacy test automation solutions. The MDTester is a collection of tools designed to address those issues. The test cases are designed using a graphical notation called UTML, that combines concepts of the UML Testing Profile (UTP) with known-patterns in test design.

1 Introduction

1.1 Architecture

The MDTester is based on the TOPCASED environment, which runs on the popular modular plugins architecture provided by the Eclipse IDE. This not only facilitates the integration of test design activities with general system design activities, but also ensures that the developed solutions can benefit from the large variety of proposed tools in that environment.

1.2 Methodology

MDTester guides test designers through the whole process from the analysis of system and user requirements, all the way down to executable test scripts. A large collection of wizards and guards are provided to ensure that errors and conceptual test design mistakes are avoided right-away. The result is a test design process that is simple and intuitive, but yet powerful enough to express complex

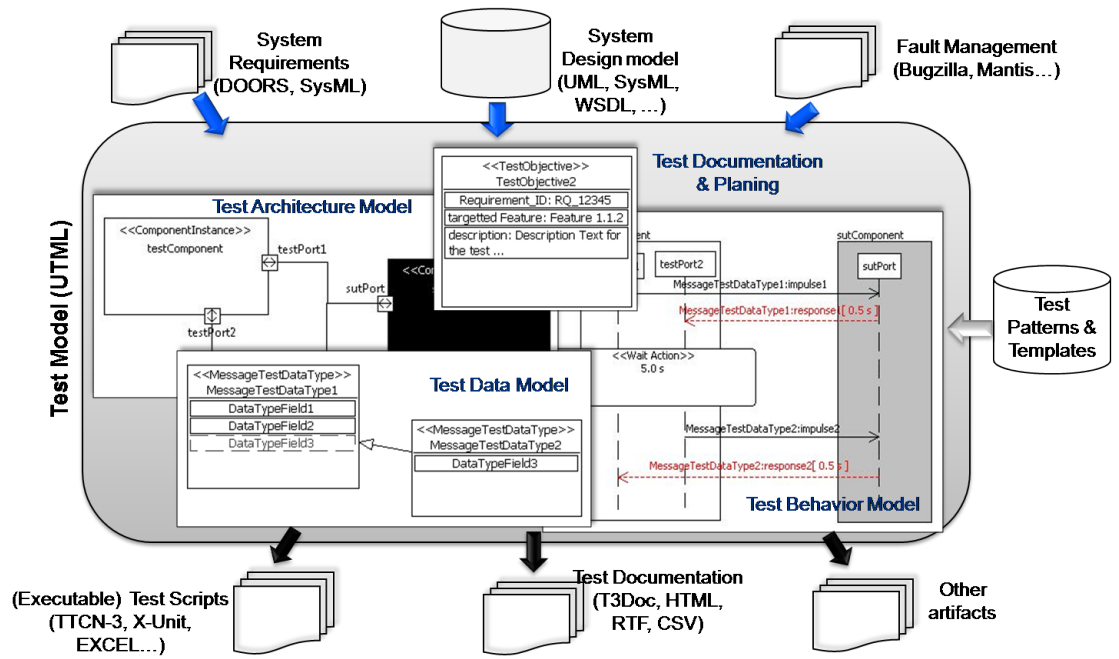


Fig. 1. MDTester Architecture

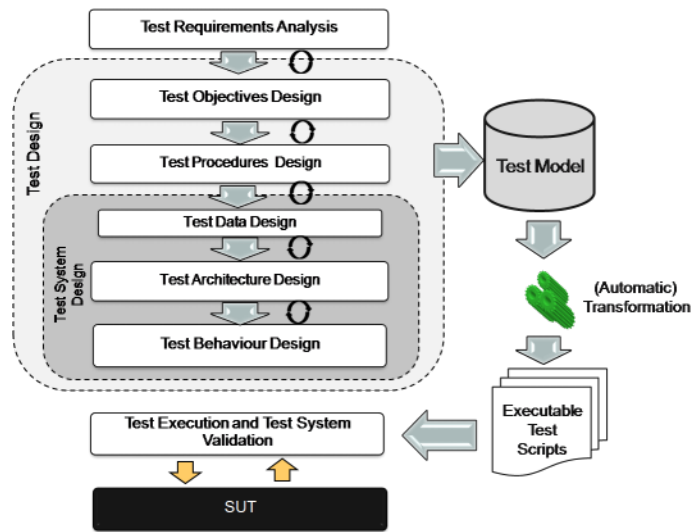


Fig. 2. MDTester Test Design Process

test behaviours. The UTML's graphical notation is simplified form of the UML notation, enhanced with specificities of test design. This ensures that the learning curve is kept low and makes it easy for people with less technical expertise, not just to understand test models, but also to design some themselves, without having to learn the details of a specific programming or scripting language.

2 Features

2.1 Test Design

MDTester supports all diagrams defined by the UTML notation to support the process depicted in Figure 2.

2.2 Test Model Validation

MDTester defines more than 50 built-in OCL-Validation rules that can be activated/deactivated as wished to verify that the created test models are valid and self-consistent. Additionally to those built-in rules, more validation rules can be added to the tool to perform quality assessment on the test models (e.g. based on company guidelines or similar)

2.3 Round-Trip Engineering

MDTester provides several back-ends and front-ends to allow round-trip engineering from PST models to executable test scripts and backwards. Additionally to the back-ends provided per default (TTCN-3, JUnit), MDTester defines an API through which customized back-ends and front-ends can be connected and used to generate proprietary test scripts or specific scripts appropriate to a given test execution environment

2.4 Requirements Traceability

SysML Requirements can be referenced in UTML test models via test objectives to provide traceability from test design to requirements and backwards.

2.5 Architecture Traceability

MDTester combines model-driven testing techniques with model-based automated test generation techniques (MBT) to transform system architectures into test architectures. This allows traceability of test cases back to the elements of the system architecture which they address and a quick identification of which test cases cover a given part of the system under test.

3 Contact Us

MDTester is available for download at <http://www.fokus.fraunhofer.de/go/utml> You may also contact us for more information via that site.