End-to-End Testing Automation in TTCN-3 environment using Conformiq Qtronic™ and Elvior MessageMagic

A Case study: Automated Testing of X-Lite SIP Softphone

EXECUTIVE SUMMARY

TTCN-3 (Test and Test Control Notation version 3) is a programming language for developing tests in telecommunications domain. **SIP** (Session Initiation Protocol) is a key protocol standard in the modern convergent telecommunications systems. **UML** (Unified Modeling Language) is one of the most widely used modeling notations for describing requirements and design beyond natural language. **Conformiq Qtronic** is a tool for generating executable tests automatically from high-level system models, developed by **Conformiq Inc.** **MessageMagic** is a TTCN-3 test development and execution platform developed by **Elvior Ltd**.

This case study demonstrates how model-based testing approach can be applied in TTCN-3 environment to achieve full testing automation, from test design to test execution. Using commercial tools for automated test design such as Conformiq Qtronic and Elvior MessageMagic for automatic execution of these test cases, TTCN-3 driven test environment can now take full advantage of end-to-end test automation.

We tested a publicly available SIP softphone, X-Lite from CounterPath, using TTCN-3 as testing language. The testing involved automatically generating TTCN-3 test cases using Conformiq Qtronic and executing those test cases using MessageMagic.

We started by creating a system model, based on SIP specifications, in UML and Java compatible notation action language. This model is required by Conformiq Qtronic in order to automatically generate TTCN-3 test suite. Then we implemented a system adapter, needed to connect MessageMagic TTCN-3 execution environment to CounterPath's X-Lite softphone, and executed this test suite against the X-Lite softphone using MessageMagic.

Keywords: Model-Based Testing, TTCN-3, SIP, UML, Automated Test Design

Introduction

In this case study we explain how we tested a publicly available SIP softphone using **end-to-end test automation** in a TTCN-3 environment. **TTCN-3** (**Test and Test Control Notation version 3**) is a notation for describing executable test cases, standardized by **ETSI** (**European Telecommunication Standards Institute**). TTCN-3 is widely applied in the telecommunications domain. Using commercial tools Conformiq Qtronic for automated test design and Elvior MessageMagic for automatic execution of these test cases, TTCN-3 driven test environment can now take full advantage of end-to-end automated testing process.

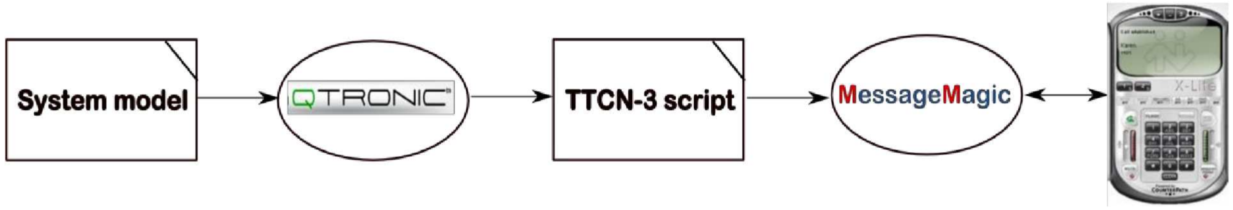
SIP stands for **Session Initiation Protocol**. It is a standardized protocol (standards body **IETF**, the **Internet Engineering Task Force**) for establishing, managing and terminating media sessions and handling the necessary infrastructure such as registering the terminals that are available for users. It is widely used, for example, in the **VoIP** (Voice over Internet Protocol) and **IMS** (IP Multimedia Subsystem).

Automated test design refers to test design automation where executable tests are automatically generated from the high-level **system models**. It extends the notion of test automation from mere test **execution** to also test **design**. Model-based automated test design reduces quality assurance risks and costs, because it removes possible human errors from the areas of **test design**, **test selection and test script coding**. Furthermore, the models created are test assets and shared with system designers and improve communication among designers and testers. This results in improved system documentation and resulting product quality.

In this case study our aim is to demonstrate how end-to-end test automation works in a TTCN-3 environment.

Automated Test Generation and Execution Workflow

We generated the tests from the SIP softphone model that presents the expected and correct behaviour of the system under test (SUT) from an external perspective. The test suite was generated using Conformiq Qtronic 2.0.3, which is available as an Eclipse plug-in. Conformiq Qtronic imports the system model of the SUT and generates test scripts using configurable back-ends, in the test language of choice. Here we used TTCN-3 back-end scripter to generate test scripts in TTCN-3 format. The test scripts were executed against the SUT, CounterPath's X-Lite SIP based softphone (version 3.0 build 53117), using MessageMagic test execution tool.



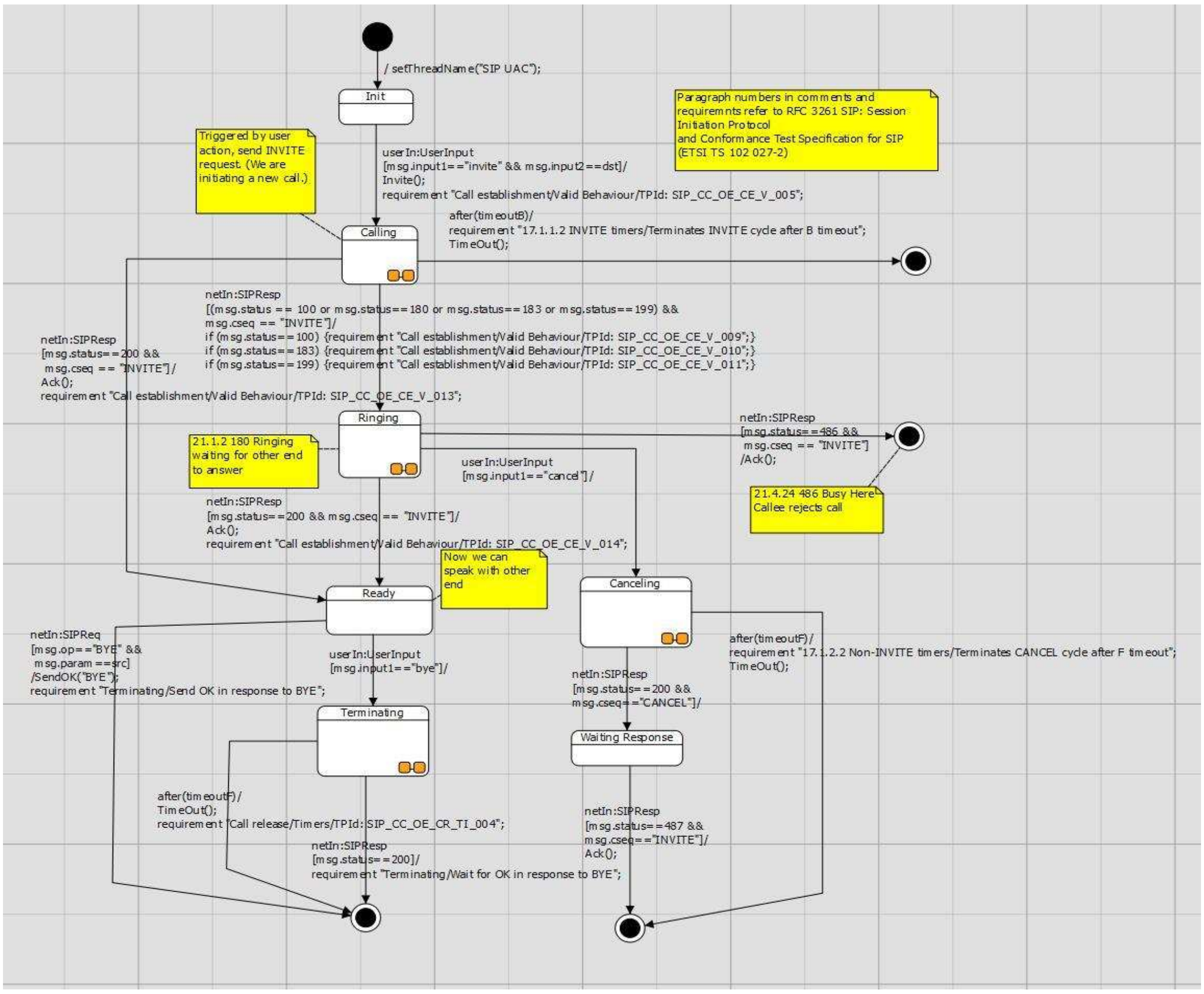
End-to-End testing automation

SUT model

We constructed the SUT model using UML 2.0 state machine and Java compatible notation action language. Conformiq Modeler, a free lightweight modelling tool from Conformiq, was used for creating the system model UML state machine. The action language artefacts were written using a standard Eclipse Java editor. Users can use other modelling tools such as Enterprise Architect, Telelogic or IBM modelling tools for their modelling needs.

We created the model of the SUT by using the SIP standard (RFC 3261) specifications as a reference. In particular, the implementation (of X-Lite) was not consulted when creating the model. This ensured that the testing was independent of the implementation, which is an important aspect in conformance testing

The functional requirements of the specifications that were selected for testing in the implementation were annotated on the model of the SUT using the keyword ***requirement***. There were **twenty one** such explicitly annotated requirements. Note that the model was built using a larger number of the requirements in the SIP specification but not all of them were chosen as testing goals for this case study.


/ setThreadName("SIP UAC");
Init
Paragraph numbers in comments and requiremnts refer to RFC 3261 SIP: Session Initiation Protocol and Conformance Test Specification for SIP (ETSI TS 102 027-2)
Triggered by user action, send INVITE request. (We are initiating a new call.)
userIn:UserInput [msg.input1=="invite" && msg.input2==dst]/ Invite(); requirement "Call establishment/Valid Behaviour/TPId: SIP_CC_OE_CE_V_005";
after(timeoutB)/ requirement "17.1.1.2 INVITE timers/Terminates INVITE cycle after B timeout"; TimeOut();
Calling
netIn:SIPResp [(msg.status == 100 or msg.status==180 or msg.status==183 or msg.status==199) && msg.cseq == "INVITE"]/ if (msg.status==100) {requirement "Call establishment/Valid Behaviour/TPId: SIP_CC_OE_CE_V_009";} if (msg.status==183) {requirement "Call establishment/Valid Behaviour/TPId: SIP_CC_OE_CE_V_010";} if (msg.status==199) {requirement "Call establishment/Valid Behaviour/TPId: SIP_CC_OE_CE_V_011";}
netIn:SIPResp [msg.status==200 && msg.cseq == "INVITE"]/ Ack(); requirement "Call establishment/Valid Behaviour/TPId: SIP_CC_OE_CE_V_013";
Ringing
21.1.2 180 Ringing waiting for other end to answer
netIn:SIPResp [msg.status==486 && msg.cseq == "INVITE"] /Ack();
21.4.24 486 Busy Here Callee rejects call
userIn:UserInput [msg.input1=="cancel"]/
netIn:SIPResp [msg.status==200 && msg.cseq == "INVITE"]/ Ack(); requirement "Call establishment/Valid Behaviour/TPId: SIP_CC_OE_CE_V_014";
Now we can speak with other end
Ready
Canceling
netIn:SIPReq [msg.op=="BYE" && msg.param ==src] /SendOK("BYE"); requirement "Terminating/Send OK in response to BYE";
userIn:UserInput [msg.input1=="bye"]/
after(timeoutF)/ requirement "17.1.2.2 Non-INVITE timers/Terminates CANCEL cycle after F timeout"; TimeOut();
netIn:SIPResp [msg.status==200 && msg.cseq=="CANCEL"]/
Terminating
Waiting Response
after(timeoutF)/ TimeOut(); requirement "Call release/Timers/TPId: SIP_CC_OE_CR_TI_004";
netIn:SIPResp [msg.status==487 && msg.cseq=="INVITE"]/ Ack();
netIn:SIPResp [msg.status==200]/ requirement "Terminating/Wait for OK in response to BYE";

System Model for generating test cases automatically

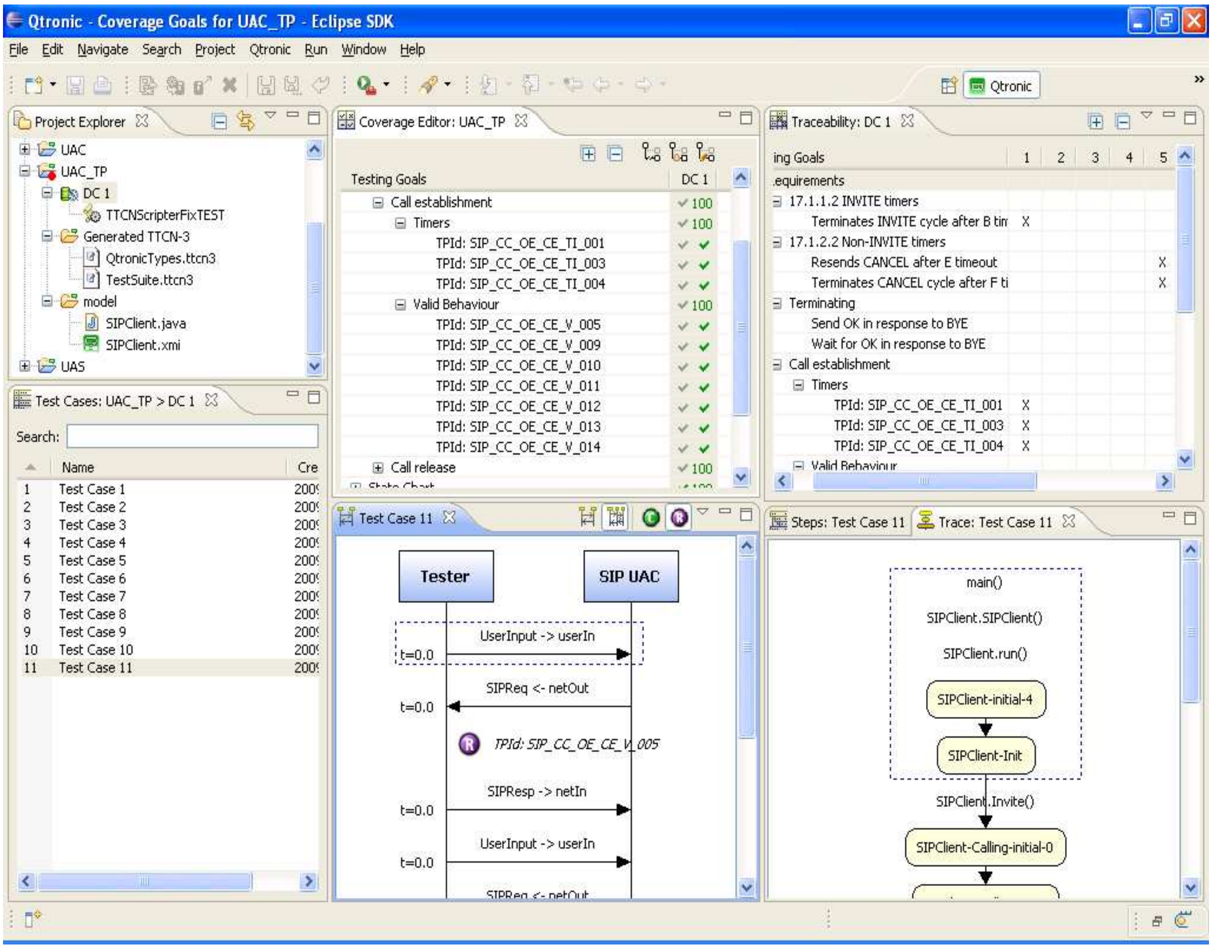
Generating the TTCN-3 Test Suite

The set of test cases is chosen automatically by Conformiq Qtronic according to defined testing goals such as requirements, states, and transitions coverage. A component of the model (for example, a method, a transition or a conditional branch) can be said to be covered by a test case if executing the test case against the SUT would guarantee that the said component of the model would be exercised during the test. Conformiq Qtronic attempts to create a test suite that covers as many of the different model components the user has selected. This approach creates a *reasonably compact* test suite that exercises a large amount of the functional features in the model.

Our goal was to generate tests that cover all annotated requirements and all states/transitions of the model. The resulting test suite contained **eleven** test cases. The generated test suite covered 100% of the annotated requirements, states and transitions.

Conformiq Qtronic provides several useful windows for analysing the generated test suites:

- **Coverage Editor** allows users to choose coverage settings. It also shows the final status of black-box coverage after the test generation cycle.
- **Test Case List** shows all the generated test cases with the date and the name. The user can rename the test cases as required.
- **Traceability Matrix View** correlates the coverage goals such as requirements to the test cases.
- **Test Case View** shows the interaction between the tester and the system under test.
- **Test Step View** shows detailed information about the messages that are transferred between the tester and system under test in the given test case.
- **Execution Trace View** links the test case to the parts of the model.

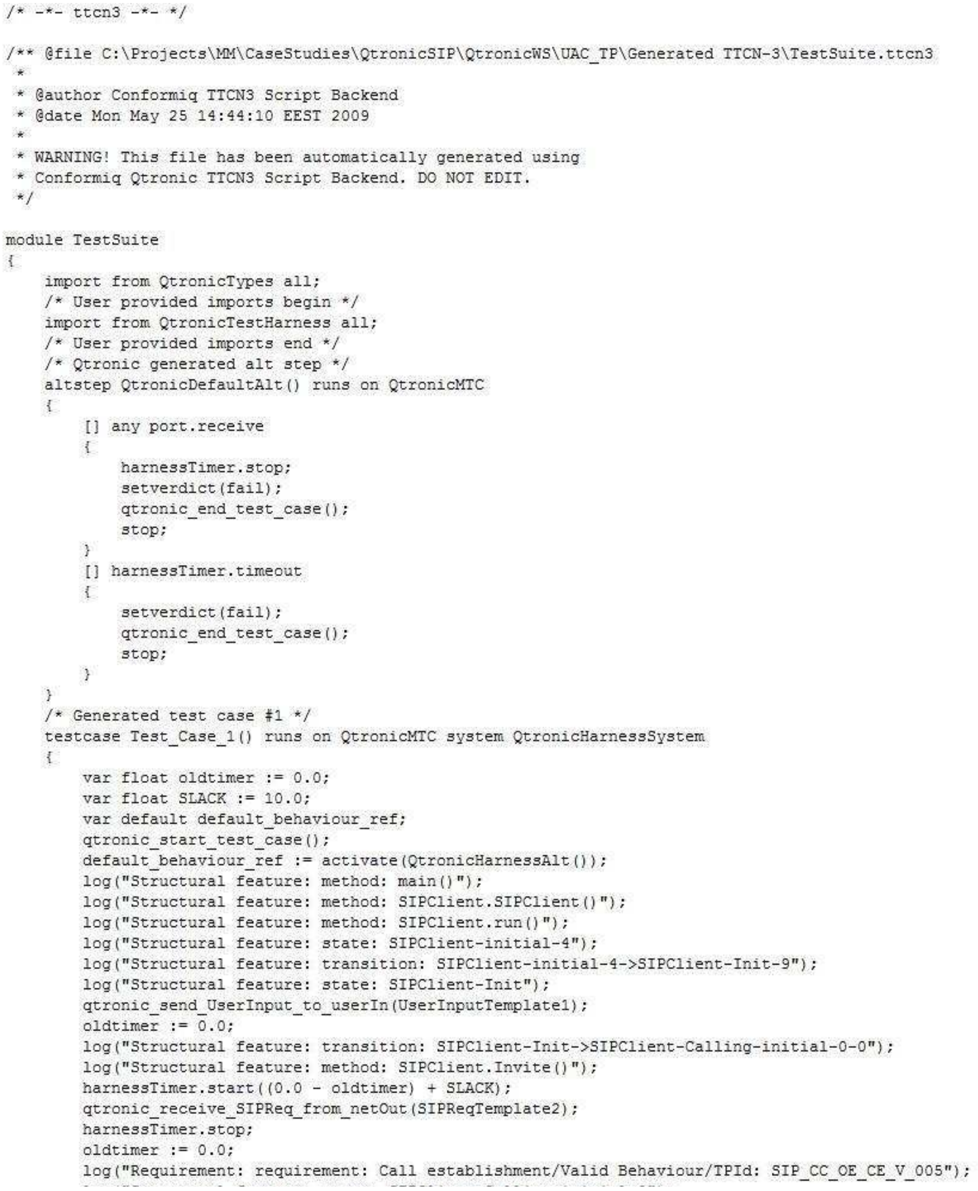


Testing Goals	DC 1
Call establishment	✓ 100
Timers	✓ 100
TPId: SIP_CC_OE_CE_TI_001	✓ ✓
TPId: SIP_CC_OE_CE_TI_003	✓ ✓
TPId: SIP_CC_OE_CE_TI_004	✓ ✓
Valid Behaviour	✓ 100
TPId: SIP_CC_OE_CE_V_005	✓ ✓
TPId: SIP_CC_OE_CE_V_009	✓ ✓
TPId: SIP_CC_OE_CE_V_010	✓ ✓
TPId: SIP_CC_OE_CE_V_011	✓ ✓
TPId: SIP_CC_OE_CE_V_012	✓ ✓
TPId: SIP_CC_OE_CE_V_013	✓ ✓
TPId: SIP_CC_OE_CE_V_014	✓ ✓
Call release	✓ 100

ing Goals	1	2	3	4	5
.equirements					
17.1.1.2 INVITE timers					
Terminates INVITE cycle after B tin	X				
17.1.2.2 Non-INVITE timers					
Resends CANCEL after E timeout					X
Terminates CANCEL cycle after F ti					X
Terminating					
Send OK in response to BYE					
Wait for OK in response to BYE					
Call establishment					
Timers					
TPId: SIP_CC_OE_CE_TI_001	X				
TPId: SIP_CC_OE_CE_TI_003	X				
TPId: SIP_CC_OE_CE_TI_004	X				

	Name	Cre
1	Test Case 1	200
2	Test Case 2	200
3	Test Case 3	200
4	Test Case 4	200
5	Test Case 5	200
6	Test Case 6	200
7	Test Case 7	200
8	Test Case 8	200
9	Test Case 9	200
10	Test Case 10	200
11	Test Case 11	200

Conformiq Qtronic test generation and analysis environment



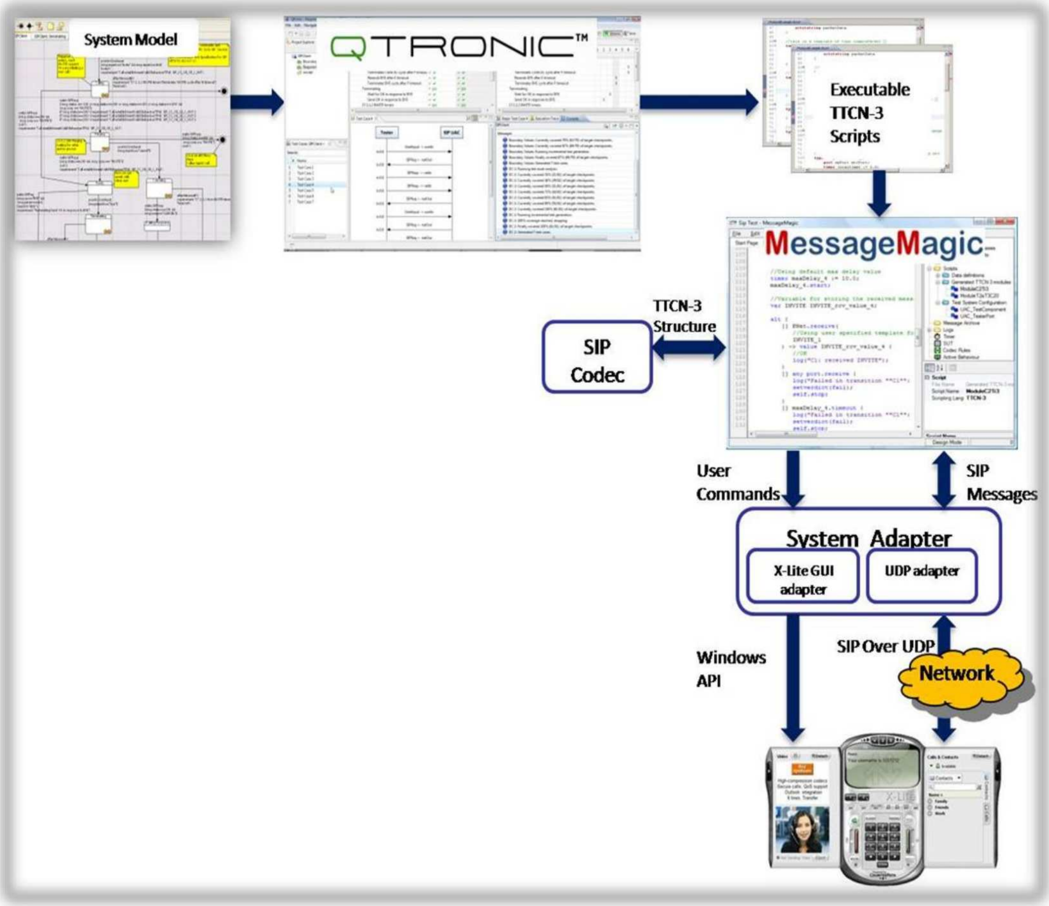
```
/* -*- ttcn3 -*- */

/** @file C:\Projects\MM\CaseStudies\QtronicSIP\QtronicWS\UAC_TP\Generated TTCN-3\TestSuite.ttcn3
 *
 * @author Conformiq TTCN3 Script Backend
 * @date Mon May 25 14:44:10 EEST 2009
 *
 * WARNING! This file has been automatically generated using
 * Conformiq Qtronic TTCN3 Script Backend. DO NOT EDIT.
 */

module TestSuite
{
    import from QtronicTypes all;
    /* User provided imports begin */
    import from QtronicTestHarness all;
    /* User provided imports end */
    /* Qtronic generated alt step */
    altstep QtronicDefaultAlt() runs on QtronicMTC
    {
        [] any port.receive
        {
            harnessTimer.stop;
            setverdict(fail);
            qtronic_end_test_case();
            stop;
        }
        [] harnessTimer.timeout
        {
            setverdict(fail);
            qtronic_end_test_case();
            stop;
        }
    }
    /* Generated test case #1 */
    testcase Test_Case_1() runs on QtronicMTC system QtronicHarnessSystem
    {
        var float oldtimer := 0.0;
        var float SLACK := 10.0;
        var default default_behaviour_ref;
        qtronic_start_test_case();
        default_behaviour_ref := activate(QtronicHarnessAlt());
        log("Structural feature: method: main()");
        log("Structural feature: method: SIPClient.SIPClient()");
        log("Structural feature: method: SIPClient.run()");
        log("Structural feature: state: SIPClient-initial-4");
        log("Structural feature: transition: SIPClient-initial-4->SIPClient-Init-9");
        log("Structural feature: state: SIPClient-Init");
        qtronic_send_UserInput_to_userIn(UserInputTemplate1);
        oldtimer := 0.0;
        log("Structural feature: transition: SIPClient-Init->SIPClient-Calling-initial-0-0");
        log("Structural feature: method: SIPClient.Invite()");
        harnessTimer.start((0.0 - oldtimer) + SLACK);
        qtronic_receive_SIPReq_from_netOut(SIPReqTemplate2);
        harnessTimer.stop;
        oldtimer := 0.0;
        log("Requirement: requirement: Call establishment/Valid Behaviour/TPId: SIP_CC_OE_CE_V_005");
```

Automatically generated TTCN-3 test case

Test Execution Environment

Architecture of the testing environment is shown in the figure below. MessageMagic TTCN-3 test execution environment was used for executing the test suites generated by Conformiq Qtronic. The system under test – X-Lite SIP softphone - is a Windows application. In order to connect the test execution tool to the SUT, a system adapter was built.

The system adapter mediated SIP messages between UDP transport and MessageMagic and user actions from MessageMagic to X-Lite graphical interface using TTCN-3 standardized TRI interface (Test Runtime Interface). The system adapter contains two adapters – UDP adapter and X-Lite GUI adapter. The UDP adapter is responsible for transferring SIP messages and it is configured to map the TTCN-3 ports to the UDP ports. The X-Lite GUI adapter is designed for eliminating the needs for user manual intervention during the test cases execution. It simulates user actions via Windows API allowing full automation of the test case execution.



End-to-End Test Automation of X-Lite softphone in TTCN-3 Environment

By the definition, here the system model is an abstracted presentation of the system from external behaviour perspective. It does not describe all the details of the system but only the essential aspects under consideration. Thus it is a golden abstract implementation of the system, excluding internal implementation details.

The generated test suite in TTCN-3 represented SIP softphone behaviour corresponding to the abstraction level in the model. A test case generated by Conformiq Qtronic is a sequence of inputs and expected outputs of abstract messages corresponding to the abstraction level used in modelling. Conformiq Qtronic TTCN-3 script back-end maps SUT inputs and outputs to corresponding function calls in TTCN-3. It is the responsibility of the test harness to maintain a mapping between abstract message values generated by Conformiq Qtronic and real SIP message values that are adapted to the particular test execution environment. In this case study the test harness was written in TTCN-3 and the system adapter carried real SIP messages.

In TTCN-3 language, messages are defined as data structures. The TTCN-3 executive uses the SIP codec for encoding the data structures to SIP text messages and vice versa. MessageMagic supports two options for encoding SIP messages: built-in text codec and external SIP codec using TCI interface. Here we used MessageMagic built-in text codec rules for mapping SIP message structures in TTCN-3 to SIP text messages.

Running the Tests

We compiled the generated tests and test harness in TTCN-3 with MessageMagic and ran the tests against the X-Lite softphone. All of the test cases concluded successfully. This was an expected result, given that we were testing a stable product. Here our focus of the testing was on the basic, core functionality of chosen SIP related requirements. The tests generated did not cover all aspects of SIP, but only those we modeled. Thus we can say that model-based test automation can be used in an incremental fashion.

Conclusion: The Value of End-to-End Test Automation

The model-based, end-to-end test automation creates a direct link between design and quality assurance. It adds new value to system modeling: in addition of their intrinsic value as documentation, system models become also testing assets. This removes risks and reduces test maintenance costs because models are easier to change than the actual test scripts.

The integration of an automated test design tool such as **Conformiq Qtronic** and a TTCN-3 test execution environment like **MessageMagic** brings process-level benefits in contexts where TTCN-3 forms an integral part of the quality assurance process. Automatically generated test scripts can be stored in configuration management / version control systems, and they can be executed Independent of the test generation system.

In this case study we have demonstrated end-to-end test automation in TTCN-3 environment, using Conformiq Qtronic model-based test generation engine and MessageMagic TTCN-3 technologies, covers the whole test process from the system modeling up to the tests execution against the SUT and test results evaluation. This example shows how model-based end-to-end test automation can be employed successfully in a TTCN-3 driven quality assurance process using the tools available from Conformiq and Elvior.

About Conformiq

Conformiq Inc, a worldwide provider of software test design solutions, is advancing its vision of 'automated test design' through technical innovation driving repeatable business benefits. Conformiq's tools automate the design of functional tests for software and systems, a traditionally manual and time-consuming task.

About Elvior

Elvior is a TTCN-3 testing services and tools provider. Elvior's mission is to make embedded and distributed software testing efficient. Elvior provides user-friendly and intuitive TTCN-3 test tool MessageMagic that save customers' time and money in tests development and legacy tests maintenance. Elvior MessageMagic is applicable in wide range of industry domains like telecommunication, telematics etc.