MBT & TTCN-3 in practice: The Ericsson RCS project

Stephan Schulz
CTO
About the Ericsson RCS Project

• Goal: Investigate the feasibility of the model-based testing approach for end-to-end and conformance testing
  – Context: Interoperability testing of RCS clients and networks

• The project created and validated two models
  – A RCS **network model** created from system specifications, standards, and actual system behavior which was validated against a real SUT via TTCN-3 test execution
  – A RCS **client model** created from system specifications simulating actual XCAP and SIP network behavior, which was validated by running it against the **network model**

Focus of this presentation

Presented at SQS 2009
About Rich Communication Suite (RCS)

The RCS Initiative is the joint effort of leading industry players to speed up and facilitate the adoption of applications and services that provide an interoperable, convergent, rich communication experience based on IMS.
What RCS looks like in practice

Chat
Messaging
Social Presence
Image Share
File Transfer
Video Share
Service Capability
CS Voice/video calls
Model-Based Testing with Conformiq

• Model Based Testing (MBT) is an umbrella of approaches where computers automatically generate tests from models based on some coverage criteria.

• Conformiq Automated Test Design is an MBT approach which uses advanced technology to automatically derive and generate test scripts, documentation, and reports directly from models specified from a system perspective.

• Such models specify the correct (expected) functionality and operation of the system to be tested.
The Conformiq MBT Workflow

1. Projects with model files
2. Extracted requirements from model listed by group / hierarchy
3. Relationship between requirements and generated tests
4. High level test view for review
Scope of Testing: RCS Presence

Real World

Model

User

RCS Client

SIP

XCAP

Presence

IMS Core

PTT and other Application Servers

SIP (ISC)

IMS Core

Mobile/Fixed Network

SIP/XCAP

SIP/XCAP

userIn

userOut

sip_Callback

XCAP_Callback

test configuration

Scope of Testing: RCS Presence
The RCS network model

- Specified at a high level of abstraction using QML (UML with Java-like action language)
Example of Abstraction: processInvite()

- Model focuses on what to test – leaves out details
  - “Yes” examples: Return codes, URI, XML doc refs
  - “No” examples: call ID, sequence number, tags handling

```java
public int processInvite(SIPRequest request){
    String clientURIFromUser = request.from.addr;
    String clientURIToUser = request.to.addr;
    XDMSconfigRequest presenceDoc = getPresenceDocumentForClient(clientURIFromUser);
    if (enableAuthorization){
        if (xdms_isClientOnBuddyList(clientURIToUser, presenceDoc)) ||
            xdm_isClientOnGrantedContactsList(clientURIToUser, presenceDoc))
            return 200; // OK
        else if(xdms_isClientBlocked(clientURIToUser, presenceDoc)) {
            requirement "Presence/Network/INVITE - deny if on blocked list";
            return 403; // Forbidden
        } else if(xdms_isClientOnPoliteBlockedContactsList(clientURIToUser, presenceDoc)) {
            requirement "Presence/Network/INVITE - deny if on politely blocked list";
            return 480; // Temporarily unavailable
        } else {
            requirement "Presence/Network/INVITE - deny if client not found";
            return 404; // Not found
        }
    }
    return 200; // OK
}
```
Abstract Generated Test Visualization

- **Tester**: netConfigRequest, t=0.0
- **IMScore**: SIPRequest, t=0.0
  - SIP/Network/Register Client A
  - SIP/Network/Send OK RESPONSE form server, t=0.0
- **SIP**: SIPResponse, t=0.0
  - XCAPRequest, t=0.0
  - XCAP/Network/Process GET, t=0.0
- **XCAP**: XCAPResponse, t=0.0

```
XCAPRequest
from: sip:user_a@mt-001.ims.stp
method: GET
requestURI: ...
rootURI: ...
documentSelector: ...
path: ...
user: ...
document: directory.xml
nodeSelector: ...
requestPayload: omitted
contentType: omitted
```
/* Generated test case #2 */
testcase test_case_2() runs on CQ_MTC system RCS_TSI
{
  var float v_last_timeout := 0.0;
  var default *v_cq_default_ref;

  f_start_test_case();

  v_cq_default_ref := activate(a_cq_default());

  f_cq_send_netConfigRequest_to_management(netConfigRequestTemplate6);
  f_cq_send_SIPRequest_to_GmIn(SIPRequestTemplate7);
  f_cq_receive_SIPResponse_from_GmOut(SIPResponseTemplate8);
  f_cq_send_XCAPRequest_to_HTTPIn(XCAPRequestTemplate9);
  f_cq_receive_XCAPResponse_from_HTTPOut(XCAPResponseTemplate10);

  setverdict(pass);

  deactivate(v_cq_default_ref);
  f_cq_end_test_case();
}
Test Execution: The complete tool chain

- Fills in & completes detail of generated TTCN-3 SIP/XCAP messages
- Performs actual TTCN-3 matching operation
Conclusions

• We have introduced and compared automated versus manual test design in the context of TTCN-3
  – Test engineers specify behavioral models using only directly relevant information
  – Automatic generation of abstract TTCN-3 test cases from models
  – Handling of details is suppressed into a MBT TTCN-3 framework
  – Industrial projects indicate significant gains in productivity

• MBT means a paradigm shift in test automation
  – A higher level of test specification requires new skill set
  – However classic testing skills are still needed and important

• Approach has been validated with complex real world case study, i.e., RCS network testing