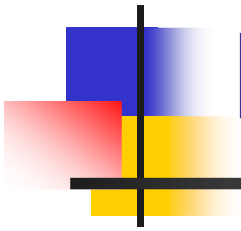


Model-Based Testing

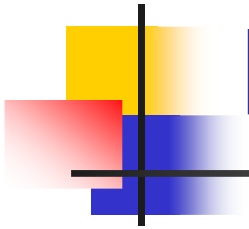
An Evaluation

Johan Nordholm
johan.nordholm24@gmail.com

Background and Motivation

- Traditional testing processes
  - Common factors:
    Manual test design & Manual test coverage analysis
- Model-based testing automates these

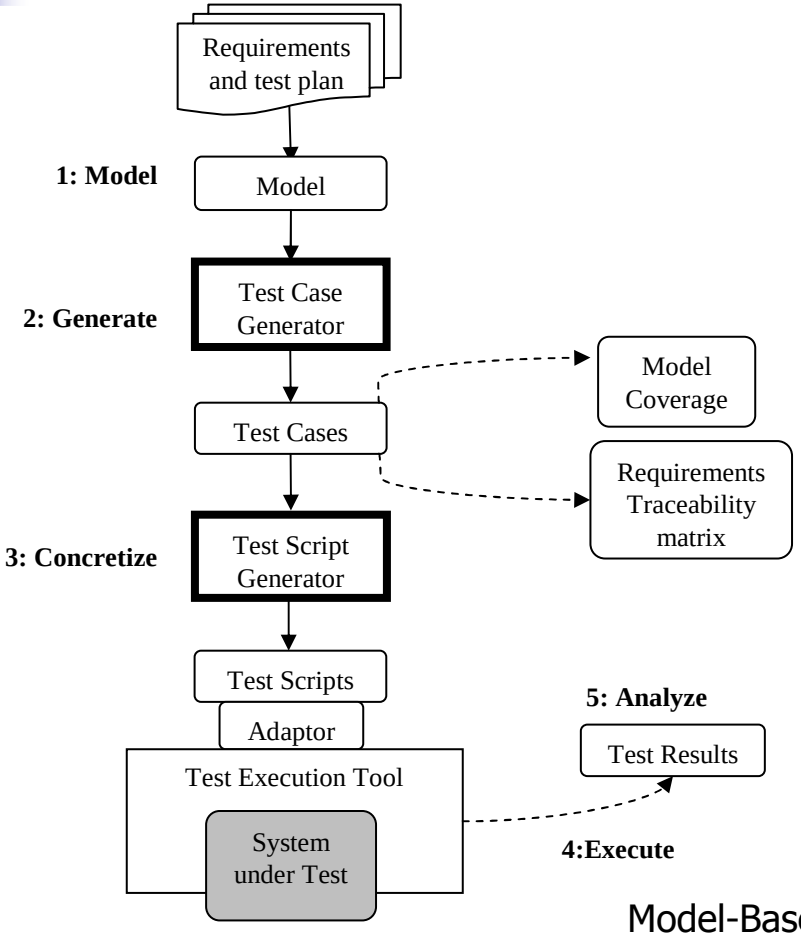
Testing Process	Test phases			
	Test cases	Test execution	Test coverage	Test result analysis
Manual	*Manual design*	*Manual execution*	*Manual analysis*	*Manual analysis*
Capture/Replay	*Manual design*	*Automated execution (records manual execution)*	*Manual analysis*	*Automated analysis (manually written)*
Script-based	*Manual design*	*Automated execution*	*Manual analysis*	*Automated analysis (manually written)*
Keyword-based	*Manual design*	*Automated execution*	*Manual analysis*	*Automated analysis (manually written)*
Model-based	***Automated design (generated from model)***	*Automated execution*	***Automated analysis (generated from model)***	***Automated analysis (generated from model)***



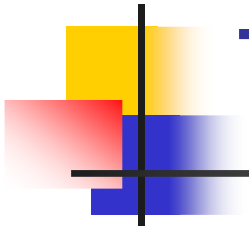
Model-Based Testing (MBT)

- **Black-box testing technique**, i.e. functional testing
- Input
  - Model of the system under test (SUT)
  - Test selection criteria
- Output
  - Derives test cases from the model, based on the test selection criteria
  - Traceability matrix, test coverage, and other test generation information
- **MBT automates**:
  - Test **case design** (generated from model)
  - Test **execution**
  - Test **coverage analysis** (generated from model)
  - Test **result analysis** (generated from model)

MBT Process



1. Model of SUT
2. Test case generation
3. Generate test code (TCL) and test harness implementation
4. Test execution
5. Test execution analysis (verdict and log)

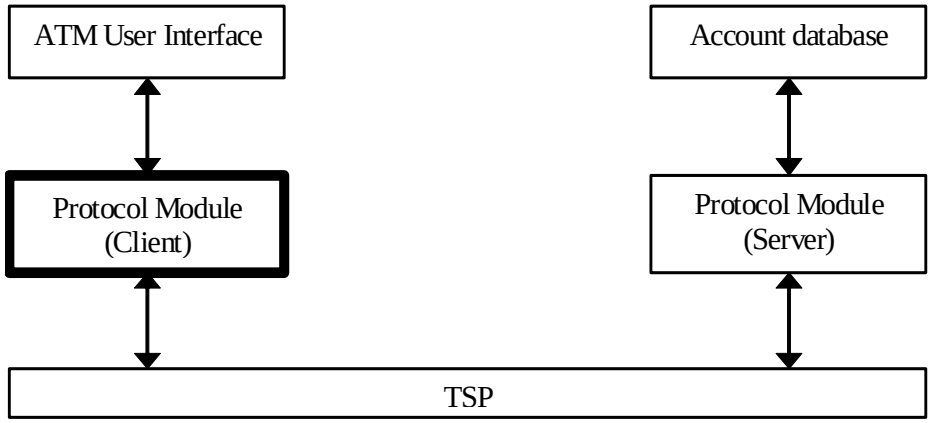


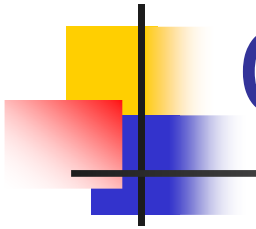
Thesis Project

- **Feasibility study of model-based testing**
  - Test object: client protocol module of ATM (Automated Teller Machine) client-server system
  - MBT tool: Qtronic
  - Evaluation: Qualitative analysis
  - 4 experiments: based on incremental development
- **Conclusions:**
  - MBT automates generation of test code, test coverage analysis and test result analysis
  - Increases level of abstraction for testing
  - Supports incremental development

Test Object

- Simplified model for an ATM (Automated Teller Machine) client-server system
- Test object limited to the protocol module of the client
  - Defined by a finite state machine
  - Authentication, account balance, withdrawal



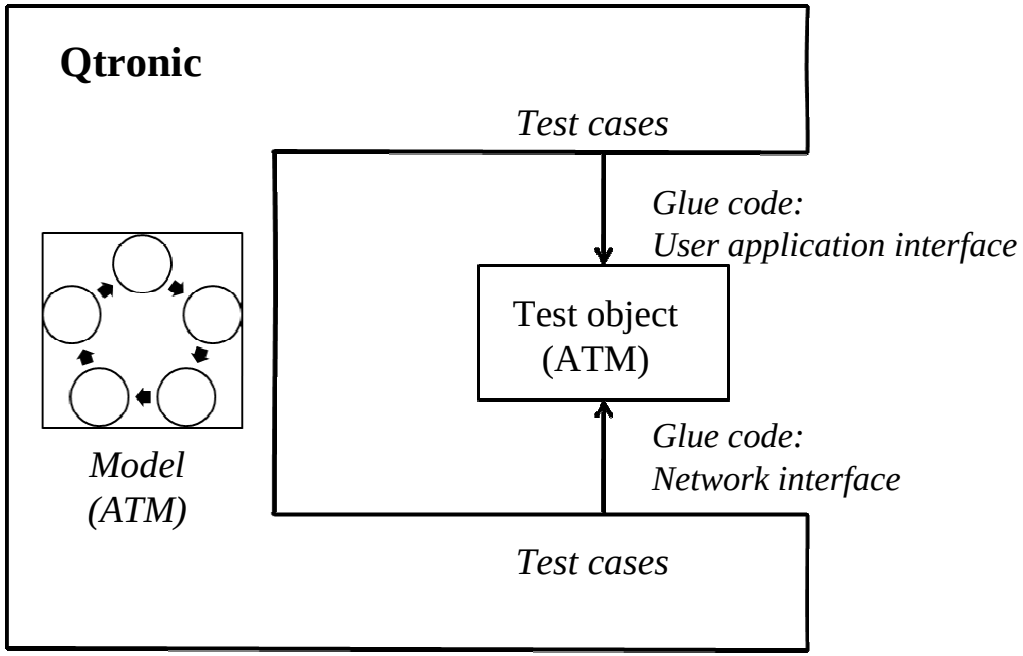


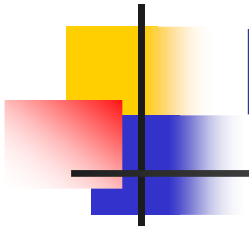
Qtronic

- **Tool for automatically designing & generating black-box tests**
  - Stand-alone version
  - Eclipse plugin
- **Input**: **Model of the system under test**
  - QML: Qtronic Modeling Language used to create models
    - Textual notation: **Based on Java**
    - Graphical notation: **Based on UML**
  - Qtronic Modeler
    - Separate modeling tool for creating graphical models
- **Output**: **Test cases and test code**
  - Generates abstract test cases
  - Generates test code from the abstract test cases

Testing with Qtronic

- Working process
  - Development of test object
  - Model development
  - Test generation and test code (TCL) generation
  - Test harness implementation (glue code) using TCL
  - Test execution





Experiments

- **Experiment 1: Initial specification**
  - Goal: Apply model-based testing on the test object and execute generated tests against that test object
- **Experiment 2: Extended specification**
  - Goal: Add requirements to see how the implication of an extended model propagates through the process
- **Experiment 3: Modified specification (authentication)**
  - Goal: Change the requirements to see how the implication of model modifications propagates through the process
- **Experiment 4: Logic implemented in test harness**
  - Goal: Evaluate the implications of moving logic from the model to the test harness

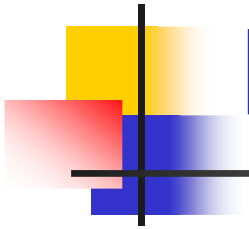
Experiment Results

- Result table for the 4 experiments

Measures	Exp 1	Exp 2	Exp 3	Exp 4
Modeling time	**2 days**	**1 day**	**4 hours**	**2 hours**
Test generation time	**13 s**	**2 min 34 s**	**3 min 11 s**	**3 m 6 s**
Test design configuration coverage	100%	100%	100%	100%
Number of generated test cases	25	52	56	56
Time to implement test harness	2 days	1 hour	10 min	2 hours
LOC: Test suite	**2860**	**6278**	**7540**	**7476**
Number of test harness procedures	18	26	28	28
LOC: Test harness	**99**	**155**	**165**	**229**
Average: LOC / Harness procedure	5.5	5.96	5.89	8.18
LOC: Test execution environment	73	73	73	73

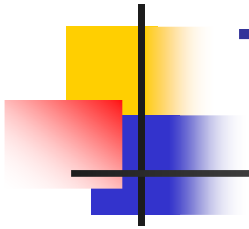
Conclusions from the Experiments

- **Modeling**
  - **The most time-consuming and challenging task**
- **Test generation**
  - **Modeling time, test generation time and generated LOC illustrate its gain**
- Test harness
  - Incrementally developed: manually (empty procedures generated)
  - Procedures for sending input and receiving output from the SUT
- Test execution environment
  - Initial effort: required for test harness implementation
- Test execution resulted in:
  - Pass/fail verdict for each test case
  - Deadlocks: model and SUT not consistent
  - Output mismatches: expected and actual output differed



Final Comments

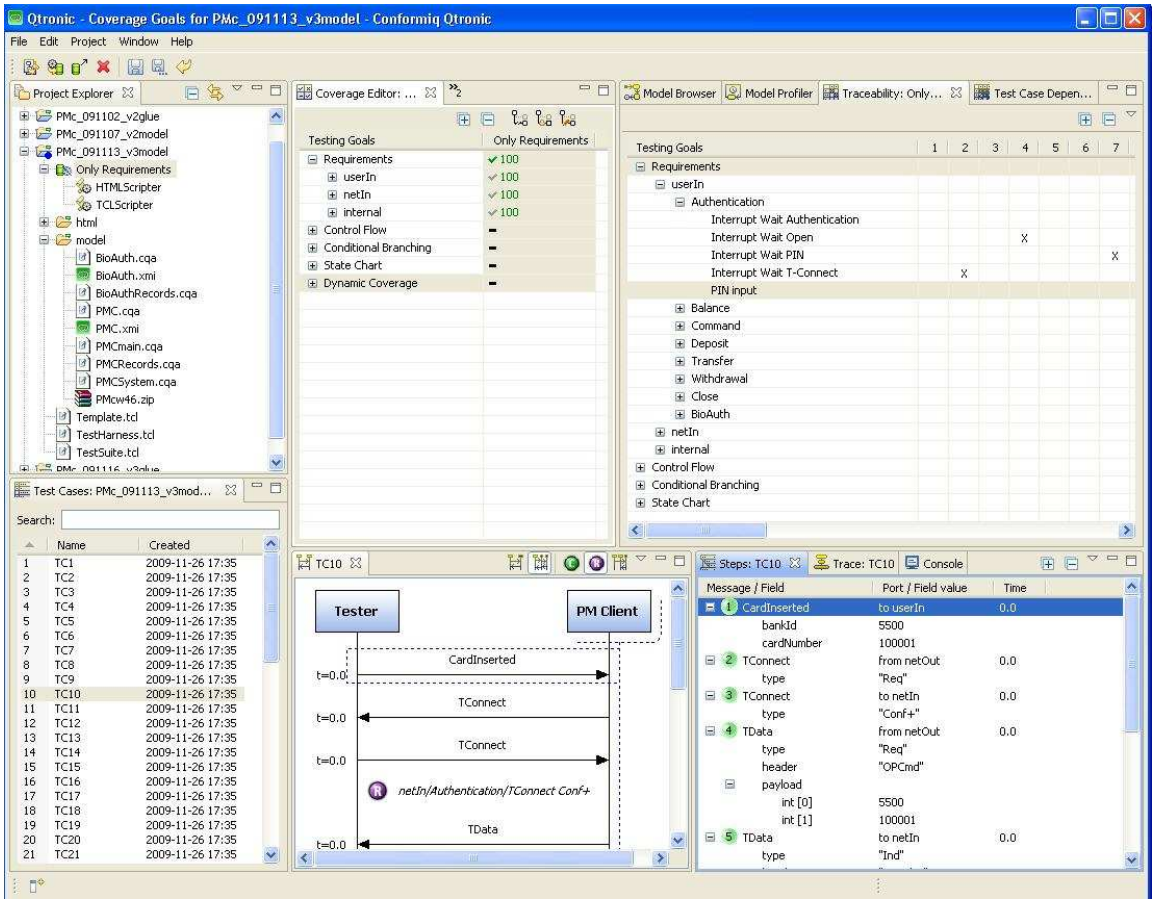
- **Conclusions**
  - **MBT automates generation of test code, test coverage analysis and test result analysis**
  - Increases level of abstraction for testing
  - Supports incremental development
  - Different skills required compared to traditional testing
- **Project a success:** evaluated the MBT concept
  - Limitation: Scalability of the project
- Model quality important
  - Model: the key testing artifact
- **Recommendation:** further evaluation of MBT



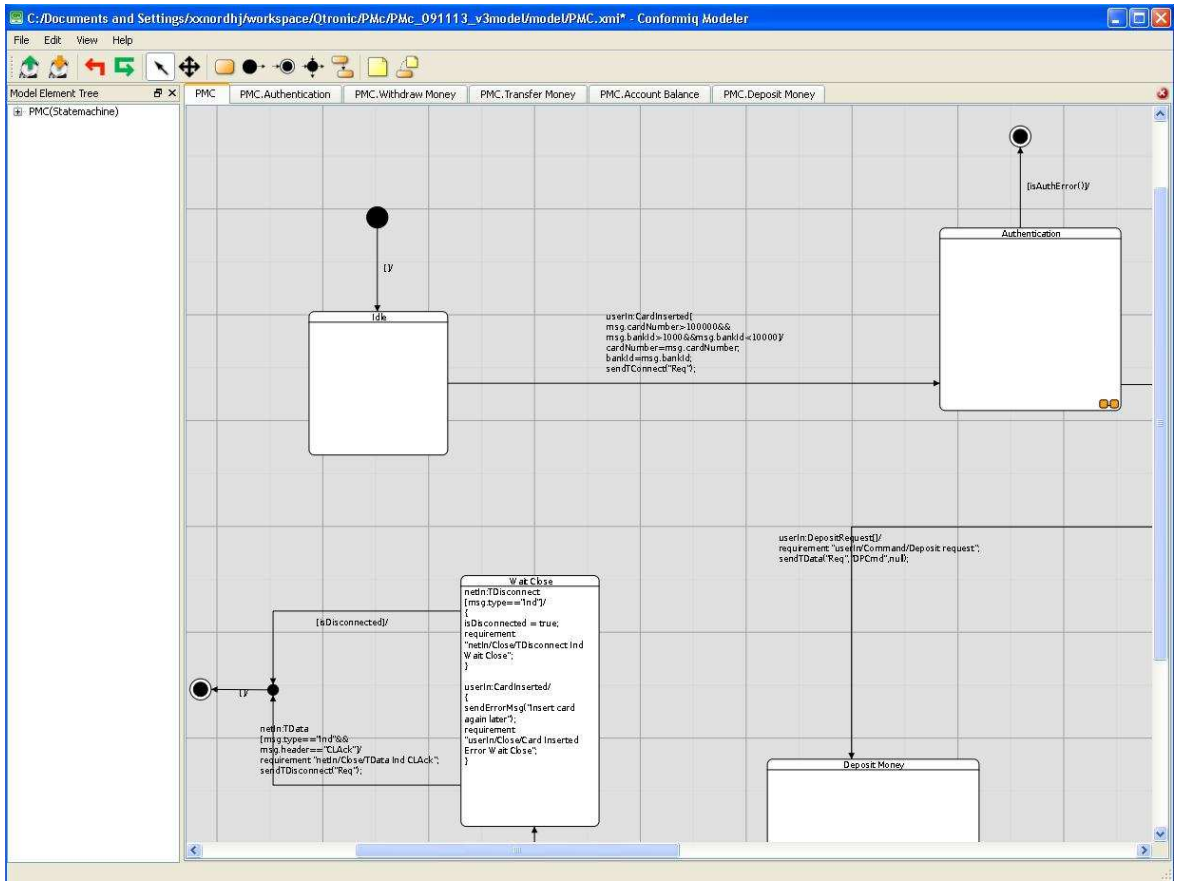
Thank You!

- **Any Questions?**
- **Contact**: johan.nordholm24@gmail.com

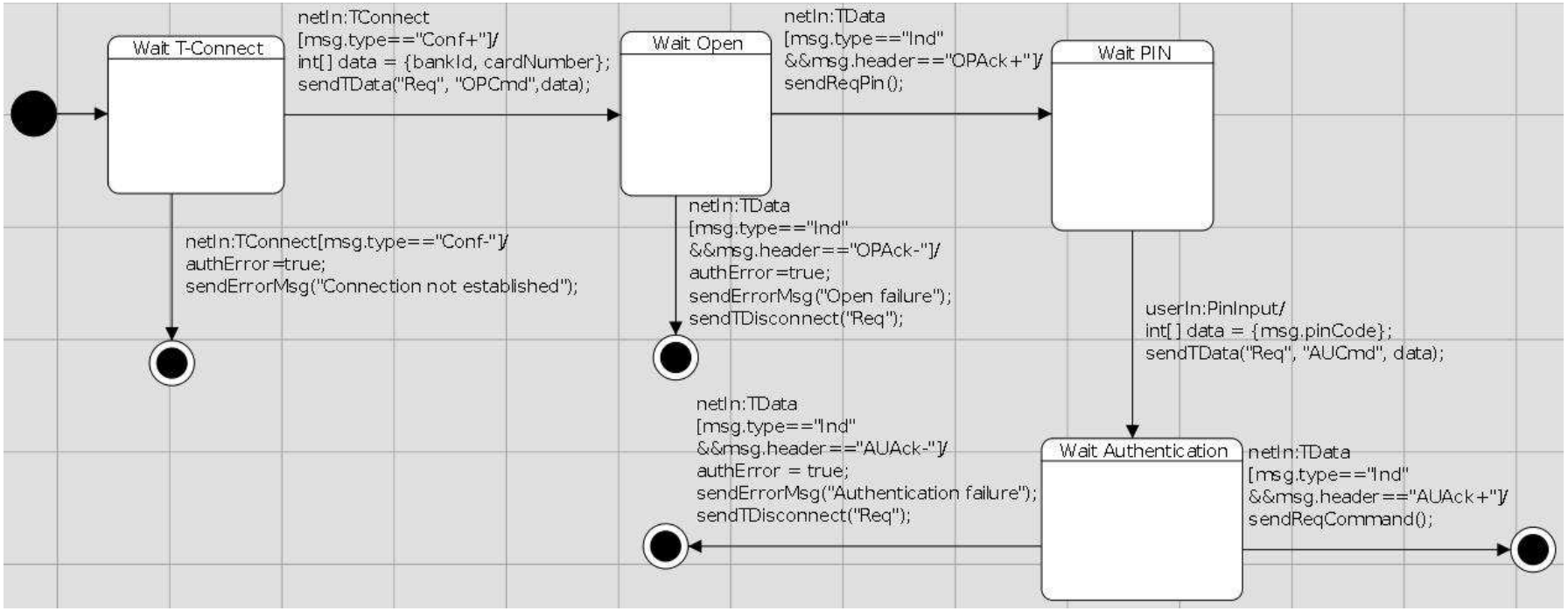
Qtronic


Qtronic - Coverage Goals for PMc_091113_v3model - Conformiq Qtronic
File Edit Project Window Help
Project Explorer
PMc_091102_v2glue
PMc_091107_v2model
PMc_091113_v3model
Only Requirements
HTMLScripter
TCLScripter
html
model
BioAuth.cqa
BioAuth.xmi
BioAuthRecords.cqa
PMC.cqa
PMC.xmi
PMCmain.cqa
PMCRecords.cqa
PMCSystem.cqa
PMcw46.zip
Template.tcl
TestHarness.tcl
TestSuite.tcl
Coverage Editor: ...
Testing Goals
Only Requirements
Requirements
userIn
netIn
internal
Control Flow
Conditional Branching
State Chart
Dynamic Coverage
Model Browser
Model Profiler
Traceability: Only...
Test Case Depen...
Testing Goals
Interrupt Wait Authentication
Interrupt Wait Open
Interrupt Wait PIN
Interrupt Wait T-Connect
PIN input
Balance
Command
Deposit
Transfer
Withdrawal
Close
BioAuth
Test Cases: PMc_091113_v3mod...
Search:
Name
Created
TC10
Tester
PM Client
CardInserted
TConnect
TConnect
netIn/Authentication/TConnect Conf+
TData
t=0.0
Steps: TC10
Trace: TC10
Console
Message / Field
Port / Field value
Time
CardInserted
to userIn
bankId
cardNumber
TConnect
from netOut
type
TConnect
to netIn
type
TData
from netOut
type
header
payload
int [0]
int [1]
TData
to netIn
type

Qtronic Modeler



QML Model


Wait T-Connect
netIn:TConnect
[msg.type=="Conf+"]/
int[] data = {bankId, cardNumber};
sendTData("Req", "OPCmd",data);
Wait Open
netIn:TData
[msg.type=="Ind"
&&msg.header=="OPAck+"]/
sendReqPin();
Wait PIN
netIn:TConnect[msg.type=="Conf-"]/
authError=true;
sendErrorMsg("Connection not established");
netIn:TData
[msg.type=="Ind"
&&msg.header=="OPAck-"]/
authError=true;
sendErrorMsg("Open failure");
sendTDisconnect("Req");
userIn:PinInput/
int[] data = {msg.pinCode};
sendTData("Req", "AUCmd", data);
netIn:TData
[msg.type=="Ind"
&&msg.header=="AUAck-"]/
authError = true;
sendErrorMsg("Authentication failure");
sendTDisconnect("Req");
Wait Authentication
netIn:TData
[msg.type=="Ind"
&&msg.header=="AUAck+"]/
sendReqCommand();

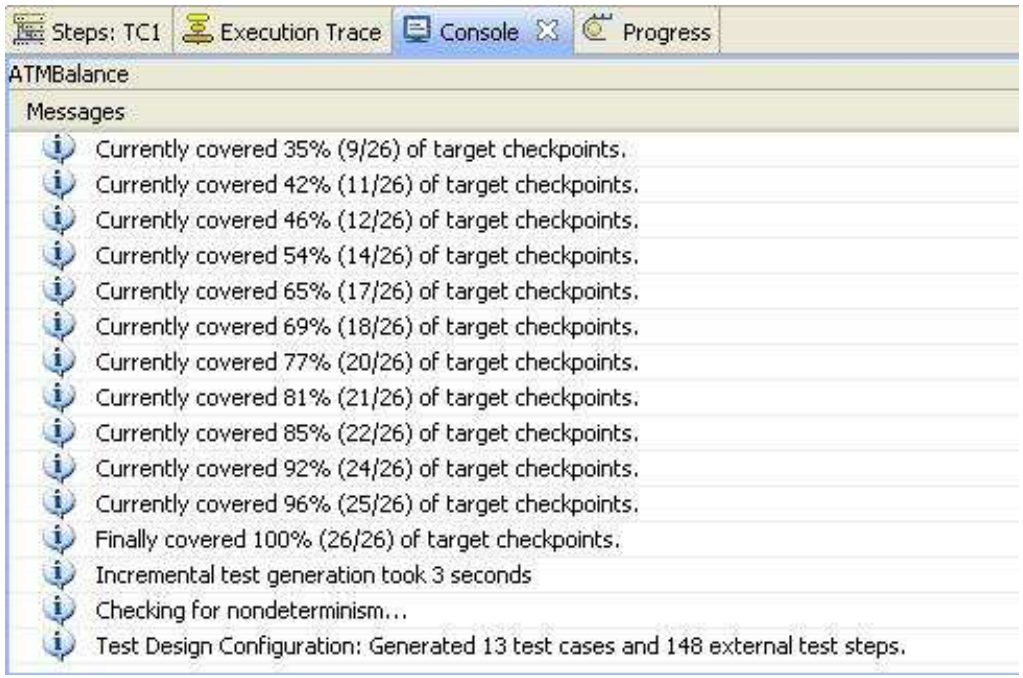
Test Design Configuration

- Example of test selection criteria in Qtronic

Testing Goals	*Test Design Configuration
⊞ Control Flow	–
⊞ Conditional Branching	–
⊟ State Chart	– 0
⊞ States	✔ 0
⊞ Transitions	–
⊞ 2-Transitions	–
⊞ Implicit Consumption	–
⊞ Dynamic Coverage	–

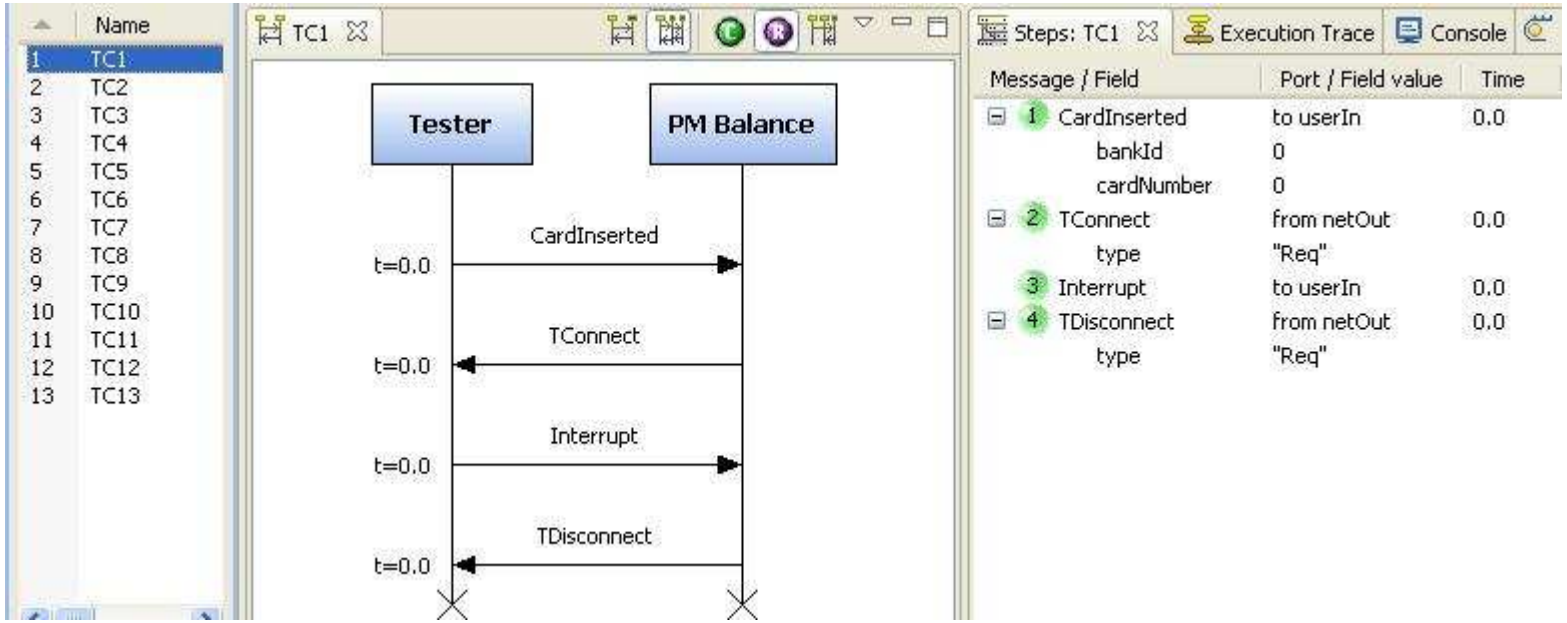
Qtronic Test Generation

- Qtronic console window



Qtronic Test Case View

- Example of the Qtronic test case view



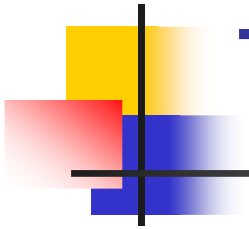
Traceability Matrix

- Traceability matrix of SIP User Agent Client example

Testing Goals	1	2	3	4	5	6	7	8	9
Requirements									
13.2.2.4 2xx Responses									
UAC core establishes session with ACK		X		X	X		X		X
15.1 Terminating a session									
UAC core terminates a session by sending BYE					X		X		X
UAS core sends OK in response to BYE				X					
17.1.1.2 INVITE timers									
Resends INVITE after A timeout	X					X			
Terminates INVITE cycle after B timeout						X			
17.1.2.2 Non-INVITE timers									
Resends BYE after E timeout							X		X
Resends CANCEL after E timeout			X					X	
Terminates BYE cycle after F timeout									X
Terminates CANCEL cycle after F timeout								X	

Test Case Example

```
proc "TC1" {} \
{
    traceprint "Running test case 'TC1'"
    traceprint "Description: This test case covers the following high level requirements:"
    traceprint " - requirement: netIn/Authentication/TConnect Conf-"
    traceprint "Action: Tester sends inbound event CardInserted to port userIn"
    set bankId_1 5500
    set cardNumber_2 100001
    userInCardInserted $bankId_1 $cardNumber_2
    traceprint "Action: SUT is expected to response with outbound event TConnect from port netOut"
    set type_3 "Req"
    netOutTConnect $type_3
    traceprint "Action: Tester sends inbound event TConnect to port netIn"
    set type_4 "Conf-"
    netInTConnect $type_4
    traceprint "Covered requirement: requirement: netIn/Authentication/TConnect Conf-"
    traceprint "Action: SUT is expected to response with outbound event ErrorMsg from port userOut"
    set msg_5 "Error message: Connection not established"
    userOutErrorMsg $msg_5
}
```

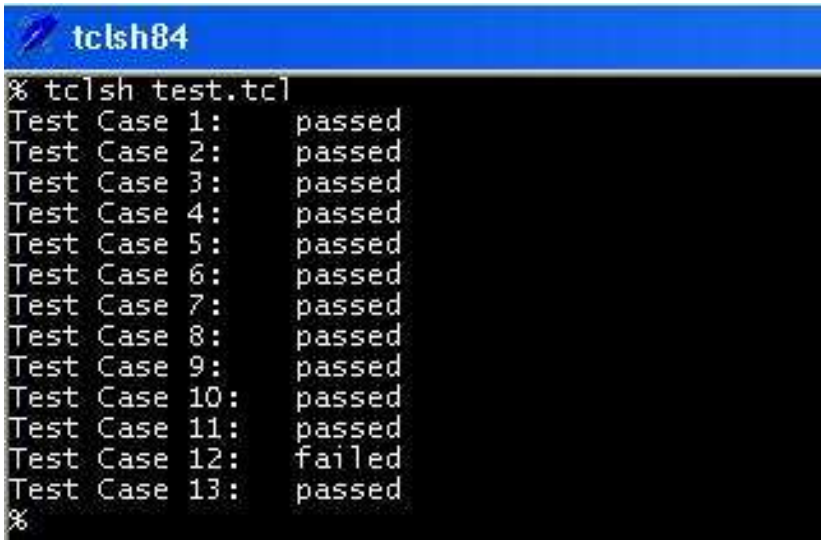


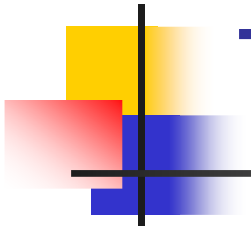
Test Harness Example

```
proc userOutDepositInfo { money } { \
      global message expected received
      set expected "Deposit info / $money"
      vwait received
}

proc userInAmountInput { money } { \
      global sockChan
      set msg "Amount input / $money"
      send $sockChan $msg
}

proc netInTDisconnect { type } { \
      global sockChan
      set msg "T-Disconnect $type"
      send $sockChan $msg
}
```

Test Execution

- Execution of test cases





Test Execution Log

Running test case 'TC1'
Description: This test case covers the following high level requirements:
 - requirement: netIn/Authentication/TConnect Conf-

Action: Tester sends inbound event CardInserted to port userIn
 +SUT input: Card inserted / 5500 / 100001

Action: SUT is expected to response with outbound event TConnect from port netOut
 -Expected output: T-Connect Req
 -Actual output: T-Connect Req

Action: Tester sends inbound event TConnect to port netIn
 +SUT input: T-Connect Conf-
Covered requirement: requirement: netIn/Authentication/TConnect Conf-

Action: SUT is expected to response with outbound event ErrorMsg from port userOut
 -Expected output: Error message: Connection not established
 -Actual output: Error message: Connection not established

 # Test Case passed