Model Based Testing for VoIP Phones

Dinesh Patil, Avaya
Jani Koivulainen, Conformiq
Jagadish Vellanki, Ideabytes

September 2010
What is Model Based Testing?

- The basic idea in Model Based Testing (MBT) is to check the conformance of a system implementation (SUT) to the specification by modeling its expected behavior.
- Directly from this model, the user selects the algorithms to use in the Automated Test Design (ATD) tool.
- ATD automatically designs and generates black box feature test scripts and proper test outcomes (test oracle).
Why Avaya Decided to Use MBT

• We realized that improving our testing by just bringing in new people was no longer an option, so we had to change our test design methodology

• MBT (Model Based Testing) and ATD (Automated Test Design) seemed like a good way to significantly boost our testing capability

• Conformiq Designer™ software was selected

• Within six months of implementation we established MBT /ATD as an essential part of our testing process
Productivity Gains Using MBT

- No manual selection and enumeration of test cases
- No need to produce test data
- No debugging of incorrect test cases
- Improved test cases and coverage
- Reports missed requirement tests
- Faster to update model than test scripts
- Maps test cases to requirements
- Consistent test case design
- Automatically matches tests to requirements
- No test automation backlog
Avaya Automated Testing Environment

Conformiq Designer™

Model

Tester Host
Windows Server

Groovy Backend

ATE (Groovy)

Generated Test Scripts

Test Harness

ATE Libraries

Text Backend

Textual Test Plans

Test Harness

Execution Logs

Network

System Under Test

UI

SNMP

Configure

(Phone + Network)
Our Groovy-based Automated Test Environment (ATE) is very complex and includes an extensive set of functionalities for interacting programmatically with our VoIP phones.
MBT Tool Operation

- Manually create system model
- Automatically designs test input and expected output with data and timer handling
- Generates test reports
- Renders test scripts in Groovy
Automated Test Design Generation
Generated Test Scripts

• In order to generate directly executable test cases for ATE, Conformiq Designer™ uses a Groovy scripting backend

• This Groovy-fragment shows what a typical test script looks like:

  // Define phone object i.e. the "end point"
  endpoint1 = define(type:'spice',alias:'endpoint1',
                    ip:.endpoint1_ip,backend:"pti-only",
                    spiceversion:"2.0")
  register(endpoint:endpoint1,extension:endpoint1_extension,
           password:'1234',gatekeeper:mygk1)

  // Navigate to contacts application using the phone object
  button(endpoint1,"AddressBook")

  // press "New" soft key on phone
  button(endpoint1,"SoftKey0")
## Testing Efficiency Results

<table>
<thead>
<tr>
<th>Test Area</th>
<th>No. of Manual Test Cases</th>
<th>No. of Conformiq Test Cases</th>
<th>No. of Manual Test Steps</th>
<th>No. of Conformiq Test Steps</th>
<th>Test Case Coverage Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Config</td>
<td>400</td>
<td>1440</td>
<td>5445</td>
<td>93000</td>
<td>3.60</td>
</tr>
<tr>
<td>Phone Apps</td>
<td>544</td>
<td>1360</td>
<td>6859</td>
<td>85000</td>
<td>2.50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test Area</th>
<th>Manual Effort in Hours</th>
<th>Conformiq Effort in Hrs</th>
<th>Productivity Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Config</td>
<td>608</td>
<td>120</td>
<td>5.07</td>
</tr>
<tr>
<td>Phone Applications</td>
<td>827</td>
<td>428</td>
<td>1.93</td>
</tr>
</tbody>
</table>
Automated Test Design

Benefits

- Higher Productivity in Test Design
- Improved Test Coverage
- Simplified Test Maintenance
- Model Reuse
- Simplified Test Harness
- Specification Issues Are Found Earlier
- Test Automation Backlog Is Eliminated
Thank You

Questions