Testing in Research and Industry

ME Work (University project + Industry Internship)

Pooja Rani
Testing is a process

- Time, Efforts
- Cost of bugs during Implementation phase
- Automating the testing process
Objective

• Analysis of software – From **testing** point of view

• Choose the **level** of analysis- class, method, module, package

• Perform analysis and prepare test reports
Approach

- Software failure mode and effect analysis (SFMEA)
  - SFMEA used to discover design defects
- Level of analysis – Method
- Modeling language – UML 2.0
SFMEA

- SFMEA present ways to analysis item. An item can be class, method, module or package.

<table>
<thead>
<tr>
<th>Item</th>
<th>Failure mode</th>
<th>Causes</th>
<th>Effects</th>
<th>Probability of occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item to be analysed for failure analysis</td>
<td>The failure modes of the selected item</td>
<td>Cause of the Failure</td>
<td>Effects that can be caused because of the failure</td>
<td>What is the probability of occurrence of this failure</td>
</tr>
</tbody>
</table>

- Considered Method as an item to analysis
Method-level Analysis

• Precondition Violation failure Modes of method
• Parametric Failure Mode of method
• Method Invocation order
• Post condition failure mode
SFMEA using Use Case Diagram

Requirement Analysis phase and Design phase

-University Project
Methodology

Parse Use Case
- Extract messages

Generate Message Precondition
- Generate message labels and precondition

Generate Scenario
- Generate scenarios from Path
Input Use Case

Software Failure Modes & Effect Analysis For Use-Cases

Use Case Name: ATM_Transaction
Actor: User1
Precondition: User should have valid ATM card

Home tab where various information is provided
Generate Message Precondition

<table>
<thead>
<tr>
<th>Action No</th>
<th>Action Label</th>
<th>Action Label Precondition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>requestWithdraw</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>InsertATM</td>
<td>A1,</td>
</tr>
<tr>
<td>A3</td>
<td>InputPIN</td>
<td>A1, A2,</td>
</tr>
<tr>
<td>A4</td>
<td>IF Condition1 THEN</td>
<td>A1, A2,</td>
</tr>
</tbody>
</table>
Scenario Generation

- Once we have scenario generate we can treat each message as item to be analyzed using SFMEA
- Perform different method-based analysis on each message
Limitation of use-case approach

- Assumption about input format
- Capture static information
- No sequence information about messages flow
- No information about message coverage
Automatic Test Case Generation using Sequence Diagram

Design phase and partially Development phase
-University Project
Why sequence diagram?

- Dynamic view of the system
- Top to bottom time sequence
- Captures object interaction
- Based on Model-based testing
- Test coverage
ATCG using Sequence Diagram

- Methodology Followed
- Sequence Diagram as input
- Pseudo code generation using SAX Parser
- Message Attributes Table generation
- Control Flow Graph generation
- Scenarios generation using CFG traversal
- Tools and Technologies
Methodology

1. Sequence diagram as input
2. Pseudocode generation using SAX parser
3. Message Attributes Table Generation
4. Scenarios generation using CFG traversal
5. Control Flow Graph Generation
Sequence Diagram as Input

Diagram showing the sequence of transactions involving a customer, ATM, and bank, with interactions such as `initiateMultipleTransactions()`, `withdrawCash(account number, amount)`, `getBalance(account number)`,
Sample XML

- Parse XML, extract tags
Pseudo code generation

- Based on various tags, generate pseudo-code
Message attributes table

- Extract Message attributes from SD
- Like Message Sequence No, Name, Type, Tag etc.

<table>
<thead>
<tr>
<th>Message Sequence No</th>
<th>Message Name</th>
<th>Message Type</th>
<th>InReply</th>
<th>Message Tag</th>
<th>From</th>
<th>To</th>
<th>Dependent Message List</th>
<th>Independent Message List</th>
<th>Send Event</th>
<th>Receive Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>cashCheque</td>
<td>Synchronous</td>
<td>False</td>
<td>1</td>
<td>Customer</td>
<td>customer</td>
<td>1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9</td>
<td></td>
<td>E11</td>
<td>E12</td>
</tr>
<tr>
<td>M2</td>
<td>getAmount</td>
<td>Synchronous</td>
<td>False</td>
<td>1.1</td>
<td>Bank</td>
<td>Cheque</td>
<td>1.3, 1.5, 1.6, 1.7, 1.8</td>
<td></td>
<td>E21</td>
<td>E22</td>
</tr>
<tr>
<td>M3</td>
<td>amount</td>
<td>Synchronous</td>
<td>True</td>
<td>1.2</td>
<td>Cheque</td>
<td>Bank</td>
<td>1.5, 1.6, 1.7, 1.8</td>
<td></td>
<td>E31</td>
<td>E32</td>
</tr>
<tr>
<td>M4</td>
<td>getBalance</td>
<td>Synchronous</td>
<td>False</td>
<td>1.3</td>
<td>Bank</td>
<td>Account</td>
<td>1.5, 1.6, 1.7, 1.8</td>
<td></td>
<td>E41</td>
<td>E42</td>
</tr>
<tr>
<td>M5</td>
<td>balance</td>
<td>Synchronous</td>
<td>True</td>
<td>1.4</td>
<td>Account</td>
<td>Bank</td>
<td>1.5, 1.6, 1.7, 1.8</td>
<td></td>
<td>E51</td>
<td>E52</td>
</tr>
<tr>
<td>M6</td>
<td>addInsufficientFunds</td>
<td>Synchronous</td>
<td>False</td>
<td>1.5</td>
<td>Bank</td>
<td>Account</td>
<td>1.6, 1.7, 1.8</td>
<td></td>
<td>E61</td>
<td>E62</td>
</tr>
<tr>
<td>M7</td>
<td>noteReturnedCheque</td>
<td>Synchronous</td>
<td>False</td>
<td>1.6</td>
<td>Bank</td>
<td>Account</td>
<td>1.7, 1.8</td>
<td></td>
<td>E71</td>
<td>E72</td>
</tr>
<tr>
<td>M8</td>
<td>addDebitTransaction</td>
<td>Synchronous</td>
<td>False</td>
<td>1.7</td>
<td>Bank</td>
<td>Account</td>
<td>1.8</td>
<td></td>
<td>E81</td>
<td>E82</td>
</tr>
<tr>
<td>M9</td>
<td>storePointsOnCheque</td>
<td>Synchronous</td>
<td>False</td>
<td>1.8</td>
<td>Bank</td>
<td>Account</td>
<td>1.9</td>
<td></td>
<td>E91</td>
<td>E92</td>
</tr>
<tr>
<td>M10</td>
<td>cash</td>
<td>Synchronous</td>
<td>True</td>
<td>1.9</td>
<td>Bank</td>
<td>Customer</td>
<td>1.9</td>
<td></td>
<td>E101</td>
<td>E102</td>
</tr>
</tbody>
</table>
Control Flow Graph (CFG) generation

- Create control graph based on pseudo code tags

- Traverse Graph using DFS

- Generate scenarios
Tools & Technologies used

- Visual Paradigm for Sequence diagram
- SAX parser for parsing XML file
- Grappa library for CFG
- Netbeans as IDE
- Java
Future Work

• Identifying valid and invalid scenario

• Elimination of redundant test case

• Generate Java test case automatically

• Execute test case automatically and generate test report

• Successfully generated java code for Finite State Machine(ModelJUnit)
Testing in Industry

VMware Internship – Testing in Development phase
Workspace ONE - Product

- Digital Workspace for enterprise application
- Giving IT a central place to manage user provisioning and access policy for applications.
VMware Identity Manager (VIDM)

- Provide identity and access management to users for Workspace ONE
- Secured access to corporate applications
- Across all devices and platforms
- Single portal access for employee work applications
- Conditional access control to apps based on device, network, and or user.
- Different Authentication Methods like RSA, Radius, certificates.
Challenges in testing VIDM

• Why Automation testing is difficult?

• Time to setup cluster environment- Short release cycles

• Identifying common scenario e.g Login test – User belonging to one domain should not be allowed to login with another domain and within one domain, group of users are there.

• Basic login test complexity

• Test coverage in Web and Mobile testing

• Lack of single testing environment

• Tools used – Selenium Grid, TestNG, Spring Framework
My Task

• Developing test libraries and extending test framework

• Identifying common scenario to automate

• Automate VIDM test cases like login test case

• Prepare regression test suite

• Integrate test cases in Continuous integration (CI) pipeline
Future Work

• Reducing test suite complexity

• Re-using test case- with every change in product needs modification in test cases

• Less GUI oriented automation test

• Increase test coverage

• Not enough coverage for mobile automation testing

• Better tools for Cloud Testing
Research Gap

- From the past experience of Samsung and VMware
- Mobile testing automation process is very slow
- Preparing test suite is still manual
- Cloud testing facing lots of challenges
- Model based testing is not used much in Industry
- Industries are skeptical about automation tools
No question? Thanks 😊