Research lines

Ana Paiva
apaiva@fe.up.pt

Pattern Based GUI Testing

Goals

• Aim to be a effectively applicable MBT approach in industry and to contribute to the construction of a higher quality GUIs and software systems
• For that PBGT focus on
  • Diminishing the time required to build a model
  • Increase the model abstraction
  • Build part of the model by reverse engineering
  • Test case explosion problem
  • Test only common recurrent behavior

Login or Authentication

• They have the same general behavior but slightly different implementations
**Pattern Based GUI Testing**

**User Interface Test Pattern \{Goal, V, A, C, P\}**

- **Defined by the developer**
  - **Goal**: ID of the test
  - **A**: sequence of actions to perform during test case execution
  - **V**: variable: set of pairs \{variable, inputData\} relating test data with variables of the test

- **Defined by the tester**
  - **V**: inputData: set of pairs \{variable, inputData\} relating test input data with the variables of the test
  - **C**: describes the final purpose (or why) the test should be executed
  - **P**: defines when the test can be executed

**Base UI Test Patterns**

<table>
<thead>
<tr>
<th>Icon</th>
<th>UITP</th>
<th>Set of Test Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call</td>
<td>{G_Call}</td>
<td>test the result of a call.</td>
</tr>
<tr>
<td>Find</td>
<td>{G_Found, G_notFound}</td>
<td>test searches returning and not returning values.</td>
</tr>
<tr>
<td>Input</td>
<td>{G_IV, G_IINV}</td>
<td>test for valid and invalid inputs.</td>
</tr>
<tr>
<td>Login</td>
<td>{G_LV, G_LINV}</td>
<td>test for valid and invalid authentications.</td>
</tr>
<tr>
<td>Sort</td>
<td>{G_SRTASC, G_SRTDESC}</td>
<td>test the sort for ascending and descending order.</td>
</tr>
<tr>
<td>MasterDetail</td>
<td>{G_MD}</td>
<td>test if exchanging the value of the master, the detail updates accordingly.</td>
</tr>
</tbody>
</table>

**PARADIGM language**

- Has nodes
  - User Interface Test Patterns
  - Structural nodes for structuring the model in different levels of abstraction
- Has arrows to establish the order of execution

**Show Results Form**
Pattern Based GUI Testing

Show Results Form

Overview

From scratch RE-TOOL Extracts UI Patterns from a Website

ME To build and configure a PARADIGM model of the Website

TG To generate test cases from the model

TE To execute the test cases on a GUI under test and generate test result reports

Pattern Based GUI Testing

Front-end

Reports
Pattern Based GUI Testing

PBGT framework extensions

- It is possible to add new UITP to the PARADIGM language to deal with new trends.
- It is possible to add new test case generation algorithms.
- It is possible to add new test drivers to test other software applications besides web and Android.

Community impact

- **Reusability concerns** - UI Test Patterns can be reused during the GUI modeling and testing process.
- **Reduced efforts** - when compared with other GUI modeling approaches, models can be crafted and configured in short time.
- **Goal focus** - typical Model-Based GUI Testing tools are centered in modeling the behavior of the application.
- With PBGT the focus is directed towards modeling testing goals.
- **Platform independent** - PBGT Tool can be used to model and test web applications and also mobile applications.
- **No source code is required** - PBGT Tool does not require access to the source code of the systems under test, in order to create or generate GUI models from them.
- **Low maintenance and evolutionary** - With few steps it is possible to extend the initial set of UI Test Patterns, and also to adjust current test strategies (or create new ones) to support new UI trends.
- **Simple to use** - With few knowledge on testing activities, users can start modeling and testing software in short time.

Community impact

http://www.fe.up.pt/~apaiva/pbgtwiki/doku.php
Pattern Based GUI Testing

Future Work

• Improve reverse engineering process
• Extend PARADIGM language with UI test patterns for web, mobile and for security testing
• Generate test data automatically
• Improve the usability of the tool
• Extend the environment to support traceability with requirements

Research lines

• Pattern Based GUI testing
• Reverse engineering
• Serious games
• Extract knowledge from user interactions

Reverse engineering

Extract models from dynamic exploration

Some papers

• Inês Coimbra Morgado, Ana C. R. Paiva, João Pascoal Faria
  Dynamic Reverse Engineering of Graphical User Interfaces

• Miguel Nabuco, Ana Cristina Ramada Paiva, João Carlos Pascoal Faria
  Inferring User Interface Patterns from Execution Traces of Web Applications
  in 14th International Conference Computational Science and Its Applications - ICCSA 2014, pp.311-326, 2014

• Clara Sacramento, Ana C. R. Paiva
  Web Application Model Generation through Reverse Engineering and UI Pattern Inferring
  in 9th International Conference on the Quality of Information and Communications Technology (QUATIC 2014), pp.-, 2014

• Miguel Nabuco, Ana C. R. Paiva, Rui Camacho, João P. Faria
  Inferring UI Patterns with Inductive Logic Programming
  in 8th Iberian Conference on Information Systems and Technologies, pp.-, 2013

• Inês Coimbra, Ana C. R. Paiva, João P. Faria, Rui Camacho
  GUI Reverse Engineering With Machine Learning
  in RAISE'12 Workshop on Realizing Artificial Intelligence Synergies in Software Engineering, pp.-, 2012
Research lines

• Pattern Based GUI testing
• Reverse engineering
• Serious games
• Extract knowledge from user interactions

Extract knowledge from user interactions

• Examples: Google analytics
• Problems: Difficult to analyse the results
• Solution: build a framework that allows to extract knowledge from that information