A Study on an Approach for Analysing Test Basis Using I/O Test Data Patterns

13th Apr. 2015
InSTA 2015 Graz, Austria
Tsuyoshi Yumoto, Tohru Matsuodani, Kazuhiko Tsuda
Agenda

1. INTRODUCTION

2. ISSUES OF TEST ANALYSIS IN BLACK BOX TESTING AND OVERVIEW OF THE TEST ANALYSIS METHOD

3. AN APPROACH TO DETERMINE SPEC-ITEM BY I/O TEST DATA PATTERNS

4. AN EXPERIMENT OF TEST ANALYSIS USING I/O TEST DATA PATTERNS

5. CONCLUSION
Agenda

1. INTRODUCTION

2. ISSUES OF TEST ANALYSIS IN BLACK BOX TESTING AND OVERVIEW OF THE TEST ANALYSIS METHOD

3. AN APPROACH TO DETERMINE SPEC-ITEM BY I/O TEST DATA PATTERNS

4. AN EXPERIMENT OF TEST ANALYSIS USING I/O TEST DATA PATTERNS

5. CONCLUSION
1. Introduction

Software testing in software development occupy 28% to 35% of the total software development effort in Japan.

Software testing is a key activity to ensure adequate balance among Q, C, and D.

Agenda

1. INTRODUCTION

2. ISSUES OF TEST ANALYSIS IN BLACK BOX TESTING AND OVERVIEW OF THE TEST ANALYSIS METHOD

3. AN APPROACH TO DETERMINE SPEC-ITEM BY I/O TEST DATA PATTERNS

4. AN EXPERIMENTATION OF TEST ANALYSIS USING I/O TEST DATA PATTERNS

5. CONCLUSION
2.1. Issues of Test Analysis in Black Box Testing

Development Lifecycle and Test Process

- Software testing is performed in multiple verification levels during the development life-cycle. These verification levels are depicted in the V model.
- This proposal is focused on the black box testing performed at the system-level verification, enclosed in a red frame.

Testing performed at each level depicted in the V model has a process similar to the development process.
2.1. Issues of Test Analysis in Black Box Testing (Cont.)

Test development process and Test analysis

- Three activities, test analysis, test design, and test implementation in the test process are also called **Test development process**.

![Diagram of V-model](image)

**Test analysis**

An activity of selecting and organizing items to be covered by the test.
“Test condition” is the output of test analysis.

- Test cases are derived based on test conditions.

Many items can be said “Test condition”.

- Deriving test cases for black box testing, pre-conditions and pre-inputs to execute AUT are selected and combined based on specification-items (spec-items).

Test design technique can be applied to design appropriate these combination.
2.1. Issues of Test Analysis in Black Box Testing (Cont.)

Issues of Test Analysis in Black Box testing

- The test condition, which is an output of test analysis, is a generic term for elements such as functions, transactions, quality characteristics, and structure elements.
- Therefore it is necessary to structure in order to sort out relationships between each element. That is also called test suite architecture.
- However, in research and practice, structuring of test conditions in the test analysis stays just experiences and heuristics.

Table 1. Examples of typical test analysis

<table>
<thead>
<tr>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
<th>Level 4</th>
<th>Level 5</th>
<th>Test conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print</td>
<td>Setting</td>
<td>Copies</td>
<td>-</td>
<td>-</td>
<td>In case of printing 100 copies.</td>
</tr>
<tr>
<td>Setting</td>
<td>Print properties</td>
<td>General</td>
<td>Anomaly conditions</td>
<td>Error massage</td>
<td>Message “print copies are over 99” is appeared.</td>
</tr>
</tbody>
</table>

Examples of Table 1 contain the following issues.

- The category “Setting” appears both in Level 1 and Level 2.
- The number of levels is not constant, thus there is a variance in meaning of each level.
- The expected result is not indicated in the upper row.
- The same specification is written on both upper and lower rows.
2.2. An Approach of Test Category based Test analysis

A set of rules for a test analysis method for black box testing utilising Test-Categories based on the Application Under Test (AUT) and fault knowledge have been proposed.

Overview of the method
The logical structure of a feature

Test –Category

The procedure and document format
2.2.1. The logical structure of a feature

• AUT is organised to a feature list
  – In order to use the logical structure of a feature as a guide, specification of AUT is organised to a feature list.

• Internal structure of AUT is inferred using this reference model.

• It can be used to test the feature in a MECE way. Each box in the logical structure can be a useful guide to determine the required test conditions.
2.2.1. The logical structure of a feature

AUT

- A set of Specifications of AUT is organised to a feature list.
  In order to use the logical structure of a feature as a guide, specification of AUT is organised to a feature list.

Feature List
2.2.2. Test-Category

Test-Category

- In order to have a consistent interpretation of determining test conditions, a name specialised for the AUT is put in each box of the logical structure.
- In order to ensure for clearly understanding the meaning of Test-Category, potential failures and/or faults which may arise for Test-Category are discussed.

Knowledge of AUT

Knowledge of faults experienced in the past

<table>
<thead>
<tr>
<th>Logical Structure</th>
<th>Test categories</th>
<th>Meaning (fault assumed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input adjustment</td>
<td>UI input</td>
<td>Validation rules to the input form, Screen control</td>
</tr>
<tr>
<td></td>
<td>Operation</td>
<td>Rules of screen transition, Start process</td>
</tr>
<tr>
<td></td>
<td>UI display</td>
<td>Results displayed, Output control</td>
</tr>
<tr>
<td></td>
<td>Printing</td>
<td>Printed information, Format</td>
</tr>
<tr>
<td>Conversion</td>
<td>Execution conditions</td>
<td>Business rules</td>
</tr>
<tr>
<td></td>
<td>Calculation</td>
<td>Fee calculation</td>
</tr>
<tr>
<td>Storage</td>
<td>Data search</td>
<td>Filter combinations, Search results</td>
</tr>
<tr>
<td>Management</td>
<td>Reflection</td>
<td>Data handling</td>
</tr>
<tr>
<td>Support</td>
<td>Error handling</td>
<td>Data handling reflected on other functions</td>
</tr>
</tbody>
</table>

Building consensus on the decided Test-Categories
2.2.3. The procedure and document format

Things used to determine test conditions from a test basis are defined.

The outputs are eliminated by layering elements based on the test case structure.
2.2.4. The Main benefit of this method

The Main benefit of this method

• Higher test coverage overall, delivering higher quality testing
  – By implementing the set of rules it will be easier to determine the necessary test conditions.

Our hypothesis

The following issues currently make determining the necessary test conditions difficult:
  1. Certain aspects of specification are not written if they are thought to be obvious.
  2. Specification is not completely written within single target section in a document.
     (for example: a behavior about a combination of functions)

– When many testers are involved in test development and proceed according to the procedure and document format, all of the testers can carry out their work according to the same set of rules.

The developed suite of test conditions are more comprehensive and do not contain duplicates
2.2.5. Experiments to confirm efficiency of the method

Two types of experiments have been performed

1. comparing quantity of spec-item determined by test analysis between a group introduced the method and a group not introduced the method [4].

2. Other experiment is that comparing quantity of spec-item determined by test analysis between before acquisition the method and after acquisition the method in one group [5]

- In both previous studies, in order to be introduced this method, it has been confirmed that particular spec-item's determination for deriving test cases can be done with less omission.

However, the method that has been proposed in previous studies haven’t been defined analysis steps exactly.

In this study, a set of I/O data in test execution is focused on as a comprehensive AUT model to be clear analysis steps.
Agenda

1. INTRODUCTION

2. ISSUES OF TEST ANALYSIS IN BLACK BOX TESTING AND OVERVIEW OF THE TEST ANALYSIS METHOD

3. AN APPROACH TO DETERMINE SPEC-ITEM BY I/O TEST DATA PATTERNS

4. AN EXPERIMENT OF TEST ANALYSIS USING I/O TEST DATA PATTERNS

5. CONCLUSION
3. AN Approach to Determine Spec-items by I/O test data patterns

Patterns of I/O data for test execution

In order to make sure of covering AUT enough in test analysis, a set of patterns of inputting and outputting data into/from the AUT (I/O data patterns) is focused as a comprehensive AUT model. This set of patterns is added to the test-category method to achieve analysing test basis can be more comprehensive.

The total combined patterns of inputting and outputting data into/from the AUT can be summarised into the nine patterns:

<table>
<thead>
<tr>
<th></th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Outside</td>
<td>Outside</td>
</tr>
<tr>
<td>P2</td>
<td>Outside</td>
<td>Inside</td>
</tr>
<tr>
<td>P3</td>
<td>Outside</td>
<td>Outside Inside</td>
</tr>
<tr>
<td>P4</td>
<td>Inside</td>
<td>Outside</td>
</tr>
<tr>
<td>P5</td>
<td>Inside</td>
<td>Inside</td>
</tr>
<tr>
<td>P6</td>
<td>Inside</td>
<td>Outside Inside</td>
</tr>
<tr>
<td>P7</td>
<td>Outside Inside</td>
<td>Outside</td>
</tr>
<tr>
<td>P8</td>
<td>Outside Inside</td>
<td>Inside</td>
</tr>
<tr>
<td>P9</td>
<td>Outside Inside</td>
<td>Outside Inside</td>
</tr>
</tbody>
</table>
### 3.1. I/O data patterns and Logical structure of a feature

Each pattern of I/O test data patterns is confirmed which aspect of logical structure of a feature can correspond to logically. Confirming result is described in the Table below.

<table>
<thead>
<tr>
<th></th>
<th>Input Adjustment</th>
<th>Output Adjustment</th>
<th>Conversion</th>
<th>Storage</th>
<th>Support</th>
<th>Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Outside</td>
<td>P1, P2, P3</td>
<td>P1, P2, P3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Inside</td>
<td></td>
<td>P4, P5, P6</td>
<td>P4, P5, P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Outside Inside</td>
<td>P7, P8, P9</td>
<td>P7, P8, P9</td>
<td>P7, P8, P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output</td>
<td>Outside</td>
<td>P1, P4, P7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Inside</td>
<td></td>
<td>P2, P5, P8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Outside Inside</td>
<td>P3, P6, P9</td>
<td>P3, P6, P9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.1. I/O data patterns and Logical structure of a feature

In a case of P1 data pattern

It can be analysed by I/O data patterns in test execution

It can be analysed by I/O data patterns in test execution
3.1. I/O data patterns and Logical structure of a feature

Each pattern of I/O test data patterns is confirmed which aspect of logical structure of a feature can correspond to logically. Confirming result is described in the Table below.

<table>
<thead>
<tr>
<th></th>
<th>Input Adjustment</th>
<th>Output Adjustment</th>
<th>Conversion</th>
<th>Storage</th>
<th>Support</th>
<th>Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outside</td>
<td>P1, P2, P3</td>
<td>P1, P2, P3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td></td>
<td></td>
<td>P4, P5, P6</td>
<td>P4, P5,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outside Inside</td>
<td>P7, P8, P9</td>
<td>P7, P8, P9</td>
<td>P7, P8, P9</td>
<td>P7, P8,</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outside</td>
<td>P1, P4, P7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td></td>
<td></td>
<td>P2, P5, P8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outside Inside</td>
<td>P3, P6, P9</td>
<td>P3, P6, P9</td>
<td>P3, P6, P9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 3.1. I/O data patterns and Logical structure of a feature

Each pattern of I/O test data patterns is confirmed which aspect of logical structure of a feature can correspond to logically. Confirming result is described in the Table below.

<table>
<thead>
<tr>
<th>Input</th>
<th>Input Adjustment</th>
<th>Output Adjustment</th>
<th>Conversion</th>
<th>Storage</th>
<th>Support</th>
<th>Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside</td>
<td>P1, P2, P3</td>
<td></td>
<td>P1, P2, P3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td></td>
<td></td>
<td>P4, P5, P6</td>
<td>P4, P5, P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outside Inside</td>
<td>P7, P8, P9</td>
<td>P7, P8, P9</td>
<td>P7, P8, P9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Input Adjustment</th>
<th>Conversion</th>
<th>Storage</th>
<th>Support</th>
<th>Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside</td>
<td>P1, P4, P7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td></td>
<td></td>
<td>P2, P5, P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outside Inside</td>
<td>P3, P6, P9</td>
<td>P3, P6, P9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.2. I/O data patterns and Logical structure of a feature

In a case of P1 data pattern

Spec-items that is other functions calling from the test executions.

The other function also can be analysed by I/O test data patterns.

Feature

- Input adjustment
- Output adjustment
- Conversion
- Storage

Support (Calling functions from inside)

Interaction (Calling functions from outside)
3.2. Consideration to Support and Interaction

Support and interaction are the classification to determine spec-items by paying attention calling the other function which relate with specific data input for the feature.

<table>
<thead>
<tr>
<th>Logical Structure</th>
<th>Trigger</th>
<th>Interruption</th>
<th>Resource sharing</th>
<th>Reflection to other feature</th>
<th>Interlock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support</td>
<td>○</td>
<td></td>
<td></td>
<td></td>
<td>○</td>
</tr>
<tr>
<td>Interaction</td>
<td></td>
<td></td>
<td>○</td>
<td>○</td>
<td></td>
</tr>
</tbody>
</table>
Agenda

1. INTRODUCTION

2. ISSUES OF TEST ANALYSIS IN BLACK BOX TESTING AND OVERVIEW OF THE TEST ANALYSIS METHOD

3. AN APPROACH TO DETERMINE SPEC-ITEM BY I/O TEST DATA PATTERNS

4. AN EXPERIMENT OF TEST ANALYSIS USING I/O TEST DATA PATTERNS

5. CONCLUSION
4. A experiment of test analysis using I/O test data patterns

**Demonstrate effectiveness of I/O test data patterns.**

**Objective:** To confirm the proposed approach can determine spec-items comprehensively more than test design in the real project.

**Evaluation:** Comparing spec-items quantity between test cases used for testing in a real project and test analysis result using I/O test data patterns.

An on-line mobile photo-sharing application.

Test cases using a real project.

Comparison

Test analysis result using I/O test data patterns.
4. A experiment of test analysis using I/O test data patterns

Real project test case
1. Summed up spec-items from test cases in the real project.
   • 491 cases -> 59 items for Upload, 151 cases -> 22 items for Grid view.
2. Making clear input data and output data for the features to be tested at system test level in test execution.
3. Adding input and output data information and classifying into I/O test data patterns.

For comparing, analysing test basis using test-categories and I/O test data patterns.

- Determining test-categories
- The input data and output data are classified into test-categories.
- To simulate the flow of data that is classified into test-categories during the test execution, spec-items are determined by test-categories.
- Comparing difference.
4. A experiment of test analysis using I/O test data patterns

Confirming test basis comprehensively by performing a simulation of the data I/O, spec-items can be determined. Missing spec-items are **not only combination spec-items such as classified into support or interaction**, even simple spec-items that can be classified into **Input Adjustment and Output Adjustment** is also lacking.
4. A part of spec-items that are lacked in the real project

<table>
<thead>
<tr>
<th>Feature</th>
<th>Spec-item</th>
<th>I/O test data pattern</th>
<th>The logical structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upload</td>
<td>Message that cannot be uploaded when you upload only a duplicate image</td>
<td>P1</td>
<td>Input Adjustment</td>
</tr>
<tr>
<td></td>
<td>Upload by another multiple devices with the same name image files at the same time</td>
<td>P3</td>
<td>Interaction</td>
</tr>
<tr>
<td></td>
<td>Album name for the new registration in the upload window</td>
<td>P4</td>
<td>Output Adjustment</td>
</tr>
<tr>
<td></td>
<td>Information change of uploaded images during upload</td>
<td>P9</td>
<td>Support</td>
</tr>
</tbody>
</table>
Agenda

1. INTRODUCTION

2. ISSUES OF TEST ANALYSIS IN BLACK BOX TESTING AND OVERVIEW OF THE TEST ANALYSIS METHOD

3. AN APPROACH TO DETERMINE SPEC-ITEM BY I/O TEST DATA PATTERNS

4. AN EXPERIMENT OF TEST ANALYSIS USING I/O TEST DATA PATTERNS

5. CONCLUSION
5. Conclusion

In this paper, an approach to classify and comprehensively analysed in the aspects of the test data I/O during the test execution have proposed.

Furthermore, using the test cases used in a real project, we tried to demonstrate the proposed approach. As a result, it has been confirmed that it is possible to detect a spec-item missing.

To utilise an I/O test data patterns more, we need to obtain still many samples. Then, we want the approach to apply to real project easier.
References

Thanks