Software Testing Techniques with Test Case Design Examples

What is Software Testing Technique?

Software Testing Techniques help you design better test cases. Since exhaustive testing is not possible; Manual Testing Techniques help reduce the number of test cases to be executed while increasing test coverage. They help identify test conditions that are otherwise difficult to recognize.

In this tutorial, you will learn 5 important software testing techniques:

Boundary Value Analysis (BVA)
Equivalence Class Partitioning

- Decision Table based testing.
- State Transition
- Error Guessing

Boundary Value Analysis (BVA)

Boundary value analysis is based on testing at the boundaries between partitions. It includes maximum, minimum, inside or outside boundaries, typical values and error values.

It is generally seen that a large number of errors occur at the boundaries of the defined input values rather than the center. It is also known as BVA and gives a selection of test cases which exercise bounding values.

This black box testing technique complements equivalence partitioning. This software testing technique base on the principle that, if a system works well for these particular values then it will work perfectly well for all values which comes between the two boundary values.

Guidelines for Boundary Value analysis

- If an input condition is restricted between values x and y, then the test cases should be designed with values x and y as well as values which are above and below x and y.
- If an input condition is a large number of values, the test case should be developed which need to exercise the minimum and maximum numbers. Here, values above and below the minimum and maximum values are also tested.
- Apply guidelines 1 and 2 to output conditions. It gives an output which reflects the minimum and the maximum values expected. It also tests the below or above values.

Example:

```
Input condition is valid between 1 to 10

Boundary values 0,1,2 and 9,10,11
```

Equivalence Class Partitioning

Equivalent Class Partitioning allows you to divide set of test condition into a partition which should be considered the same. This software testing method divides the input domain of a program into classes of data from which test cases should be designed.

The concept behind this technique is that test case of a representative value of each class is equal to a test of any other value of the same class. It allows you to Identify valid as well as invalid equivalence classes.

Example:

Input conditions are valid between

```
1 to 10 and 20 to 30
```

Hence there are three equivalence classes

```
--- to 0 (invalid)
1 to 10 (valid)
11 to 19 (invalid)
20 to 30 (valid)
31 to --- (invalid)
```

You select values from each class, i.e.,

```
-2, 3, 15, 25, 45
```

Decision Table Based Testing.

A decision table is also known as to Cause-Effect table. This software testing technique is used for functions which respond to a combination of inputs or events. For example, a submit button should be enabled if the user has entered all required fields.

The first task is to identify functionalities where the output depends on a combination of inputs. If there are large input set of combinations, then divide it into smaller subsets which are helpful for managing a decision table.

For every function, you need to create a table and list down all types of combinations of inputs and its respective outputs. This helps to identify a condition that is overlooked by the tester.

Following are steps to create a decision table:

- Enlist the inputs in rows
- Enter all the rules in the column
- Fill the table with the different combination of inputs
- In the last row, note down the output against the input combination.

Example: A submit button in a contact form is enabled only when all the inputs are entered by the end user.

	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
Input								
Name	F	T	F	T	F	T	F	T
Email	F	F	T	T	F	F	T	T
Message	F	F	F	F	T	T	T	T
Output								
Submit	F	F	F	F	F	F	F	T

(/images/1/053018_0554_SoftwareTes1.png)

State Transition

In State Transition technique changes in input conditions change the state of the Application Under Test (AUT). This testing technique allows the tester to test the behavior of an AUT. The tester can perform this action by entering various input conditions in a sequence. In State transition technique, the testing team provides positive as well as negative input test values for evaluating the system behavior.

Ad closed by Google

Stop seeing this ad | Why this ad?

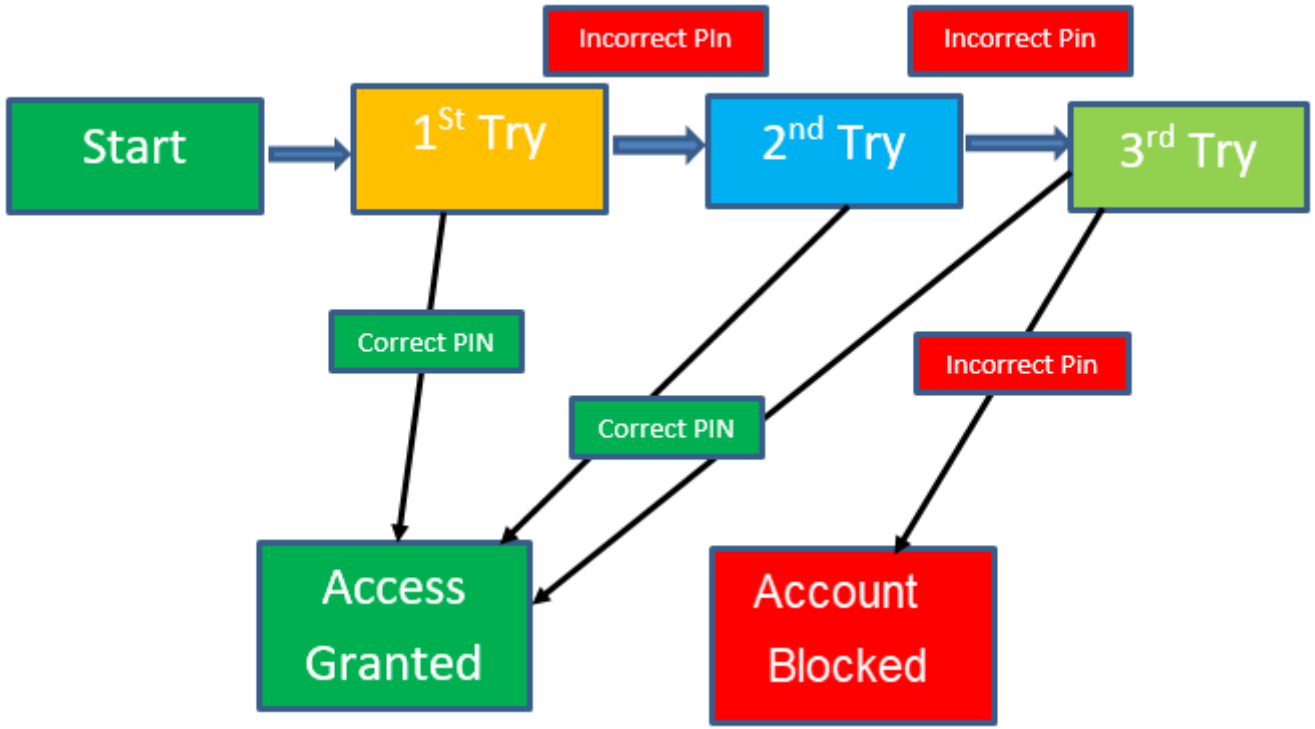
Guideline for State Transition:

- State transition should be used when a testing team is testing the application for a limited set of input values.
- The technique should be used when the testing team wants to test sequence of events which happen in the application under test.

Example:

In the following example, if the user enters a valid password in any of the first three attempts the user will be able to log in successfully. If the user enters the invalid password in the first or second try, the user will be prompted to re-enter the password. When the user enters password incorrectly 3rd time, the action has taken, and the account will be blocked.

State transition diagram



(/images/1/053018_0554_SoftwareTes2.png)

In this diagram when the user gives the correct PIN number, he or she is moved to Access granted state. Following Table is created based on the diagram above-

State Transition Table

	Correct PIN	Incorrect PIN
S1) Start	S5	S2
S2) 1st attempt	S5	S3
S3) 2nd attempt	S5	S4

S4) 3rd attempt	**S5**	**S6**
S5) Access Granted	**-**	**-**
S6) Account blocked	**-**	**-**

In the above-given table when the user enters the correct PIN, the state is transitioned to Access granted. And if the user enters an incorrect password, he or she is moved to next state. If he does the same 3rd time, he will reach the account blocked state.

Error Guessing

Error guessing is a software testing technique which is based on guessing the error which can prevail in the code. It is an experience-based technique where the test analyst uses his/her or experience to guess the problematic part of the testing application.

The technique counts a list of possible errors or error-prone situations. Then tester writes a test case to expose those errors. To design test cases based on this software testing technique, the analyst can use the past experiences to identify the conditions.

Guidelines for Error Guessing:

Ad closed by Google
Stop seeing this ad Why this ad?

- The test should use the previous experience of testing similar applications
- Understanding of the system under test
- Knowledge of typical implementation errors
- Remember previously troubled areas
- Evaluate Historical data & Test results

Conclusion

- Software testing Techniques allow you to design better cases. There are five primarily used techniques.
- Boundary value analysis is testing at the boundaries between partitions.
- Equivalent Class Partitioning allows you to divide set of test condition into a partition which should be considered the same.
- Decision Table software testing technique is used for functions which respond to a combination of inputs or events.
- In State Transition technique changes in input conditions change the state of the Application Under Test (AUT)
- Error guessing is a software testing technique which is based on guessing the error which can prevail in the code.

Next ❯

YOU MIGHT LIKE:

AGILE TESTING

(/scrum-vs-kanban.html) (/scrum-vs-kanban.html)

Scrum Vs. Kanban: Know the Difference

(/scrum-vs-kanban.html)

AGILE TESTING

(/agile-vs-kanban.html) (/agile-vs-kanban.html)

Agile Vs Kanban: What's the Difference?

(/agile-vs-kanban.html)

SOFTWARE TESTING


(/spike-testing.html) (/spike-testing.html)

What is Spike Testing? Learn With Example

(/spike-testing.html)

AGILE TESTING

(/waterfall-vs-agile.html) (/waterfall-vs-agile.html)

Waterfall Vs. Agile: Must Know Differences

(/waterfall-vs-agile.html)

SOFTWARE TESTING

(/embedded-software-testing.html)


(/embedded-software-testing.html)

What is Embedded Testing in Software Testing?

(/embedded-software-testing.html)

SOFTWARE TESTING

(/data-driven-testing.html) (/data-driven-testing.html)

What is Data Driven Testing? Learn to create Framework

(/data-driven-testing.html)

Testing Tutorials

(https://www.facebook.com/guru99com/)
(https://twitter.com/guru99com)

(https://www.youtube.com/channel/UC19i1XD6k88KqHlET8atqFQ)

(https://forms.aweber.com/form/46/724807646.htm)

About

About Us (/about-us.html)
Advertise with Us (/advertise-us.html)
Write For Us (/become-an-instructor.html)
Contact Us (/contact-us.html)

Career Suggestion

SAP Career Suggestion Tool (/best-sap-module.html)
Software Testing as a Career (/software-testing-career-complete-guide.html)

Interesting

Books to Read! (/books.html)
Blog (/blog/)
Quiz (/tests.html)
eBook (/ebook-pdf.html)

Execute online

Execute Java Online (/try-java-editor.html)
Execute Javascript (/execute-javascript-online.html)
Execute HTML (/execute-html-online.html)
Execute Python (/execute-python-online.html)

© Copyright - Guru99 2019