LECTURE 8

TEST DESIGN TECHNIQUES - I

EQUIVALENCE CLASS PARTITIONING:

- **Equivalence Class Partitioning** is a test case design technique in **black box testing**.
- Equivalence partitioning is a Test Case Design Technique to divide the input data of software into different equivalence data classes.
- Test cases are designed for equivalence data class. The equivalence partitions are frequently derived from the requirements specification for input data that influence the processing of the test object.
- A use of this method reduces the time necessary for testing software using less and effective test cases.

\[
\text{Equivalence Partitioning} = \text{Equivalence Class Partitioning} = \text{ECP}
\]

It can be used at any level of software for testing and is preferably a good technique to use first. In this technique, only one condition to be tested from each partition. Because we assume that, all the conditions in one partition behave in the same manner by the software. In a partition, if one condition works other will definitely work. Likewise we assume that, if one of the conditions does not work then none of the conditions in that partition will work.

Equivalence partitioning is a testing technique where input values set into classes for testing.

- **Valid Input Class = Keeps all valid inputs.**
- **Invalid Input Class = Keeps all Invalid inputs.**

Let's consider the behavior of tickets in the Flight reservation application, while booking a new flight.

**Example 1**

Let's consider the behavior of tickets in the Flight reservation application, while booking a new flight.

Ticket values 1 to 10 are considered valid & ticket is booked. While value 11 to 99 is considered invalid for reservation and error message will appear, "**Only ten tickets may be ordered at one time**".
**Here is the test condition**

1. Any Number greater than 10 entered in the reservation column (let say 11) is considered invalid.
2. Any Number less than 1 that is 0 or below, then it is considered invalid.
3. Numbers 1 to 10 are considered valid
4. Any 3 Digit Number say -100 is invalid.

We cannot test all the possible values because if done, number of test cases will be more than 100. To address this problem, we use equivalence partitioning hypothesis where we divide the possible values of tickets into groups or sets as shown below where the system behavior can be considered the same.

The divided sets are called Equivalence Partitions or Equivalence Classes. Then we pick only one value from each partition for testing. The hypothesis behind this technique is that if one condition/value in a partition passes all others will also pass. Likewise, if one condition in a partition fails, all other conditions in that partition will fail.
Example 2

- A text field permits only numeric characters
- Length must be 6-10 characters long

Partition according to the requirement should be like this:

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Invalid</strong></td>
<td><strong>Valid</strong></td>
<td><strong>Invalid</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

At the time of testing, test 4 and 12 as invalid values and 7 as valid one.

It is easy to test input ranges 6–10 but harder to test input ranges 2-600. Testing will be easy in the case of lesser test cases but you should be very careful. Assuming, valid input is 7. That means, you belief that the developer coded the correct valid range (6-10).

Example 3

[Diagram showing partition of EP Example - Date (1 to 31)]
Example 4

Boundary value Analysis:

- In Boundary Value Analysis, you test boundaries between equivalence partitions
- **Boundary value analysis** is a test case design techniques in **black box testing**.

- Normally Boundary value analysis is part of **stress and negative testing**.

- In our earlier example instead of checking, one value for each partitions you will check the values at the partitions like 0, 1, 10, and 11 and so on. As you may observe, you test values at **both valid and invalid boundaries**.

- Boundary Value Analysis is also called **range checking**.

- **For Example:** Using Boundary Value Analysis technique tester creates test cases for required input field. For example; an Address text box which allows maximum 500 characters. So, writing test cases for each character once will be very difficult so that will choose boundary value analysis.
Example 1

Suppose you have very important tool at office, accept valid User Name and Password field to work on that tool, and accept minimum 8 characters and maximum 12 characters. Valid range 8-12, Invalid range 7 or less than 7 and invalid range 13 or more than 13.

<table>
<thead>
<tr>
<th>Invalid Partition</th>
<th>Valid Partition</th>
<th>Invalid Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less than 8</td>
<td>8 - 12</td>
<td>More than 12</td>
</tr>
</tbody>
</table>

Write Test Cases for Valid partition value, Invalid partition value and exact boundary value.

- Test Cases 1: Consider password length less than 8.
- Test Cases 2: Consider password of length exactly 8.
- Test Cases 3: Consider password of length between 9 and 11.
- Test Cases 4: Consider password of length exactly 12.
- Test Cases 5: Consider password of length more than 12.

Example 2

Test cases for the application whose input box accepts numbers between 1-1000. Valid range 1-1000, Invalid range 0 and Invalid range 1001 or more.

<table>
<thead>
<tr>
<th>Invalid Partition</th>
<th>Valid Partition</th>
<th>Invalid Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1 - 1000</td>
<td>1001 or more</td>
</tr>
</tbody>
</table>

Write Test Cases for Valid partition value, Invalid partition value and exact boundary value.

- Test Cases 1: Consider test data exactly as the input boundaries of input domain i.e. values 1 and 1000.
- Test Cases 2: Consider test data with values just below the extreme edges of input domains i.e. values 0 and 999.
- Test Cases 3: Consider test data with values just above the extreme edges of input domain i.e. values 2 and 1001.
Equivalence partitioning and boundary value analysis are closely related and can be used together at all levels of testing.

**Example 3**

<table>
<thead>
<tr>
<th>Invalid Partition – Valid Partition Lower Boundary</th>
<th>Invalid Partition – Valid Partition Upper Boundary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boundary value just below the boundary</td>
<td>Boundary value just above the boundary</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>31</td>
<td>32</td>
</tr>
</tbody>
</table>

**Example 4**

<table>
<thead>
<tr>
<th>Invalid Partition – Valid Partition Lower Boundary</th>
<th>Invalid Partition – Valid Partition Upper Boundary</th>
</tr>
</thead>
<tbody>
<tr>
<td>BV just below the boundary</td>
<td>BV just above the boundary</td>
</tr>
<tr>
<td>5 characters</td>
<td>6 characters</td>
</tr>
<tr>
<td>10 characters</td>
<td>11 characters</td>
</tr>
</tbody>
</table>

**Example 5**

<table>
<thead>
<tr>
<th>Invalid Partition – Valid Partition Lower Boundary</th>
<th>Invalid Partition – Valid Partition Upper Boundary</th>
</tr>
</thead>
<tbody>
<tr>
<td>BV just below the boundary</td>
<td>BV just above the boundary</td>
</tr>
<tr>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>0.8</td>
<td>0.9</td>
</tr>
</tbody>
</table>