Measuring Testing Adequency

by
Ramraj S ME.,
Assistant Professor
Department of Software Engineering

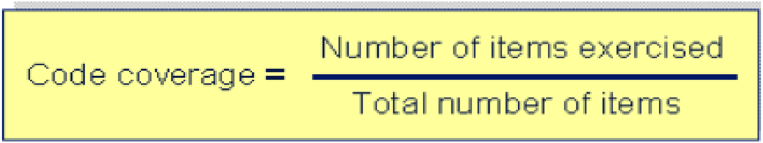
18 -Feb -2017

Agenda

- Code Coverage
- Statement Coverage
- Branch Coverage
- Relation Between Statement and Code Coverage

Code Coverage

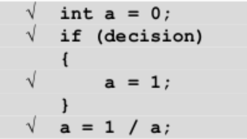
Code coverage is based on the control-structure of a piece of software respectively the flow of control achieved by the execution of a test case for that piece of software.



$$\text{Code coverage} = \frac{\text{Number of items exercised}}{\text{Total number of items}}$$

Figure : Coverage

Statement Coverage

- Statement coverage is sometimes called line coverage
- Covers all executable Statements
- Weak Coverage
- 

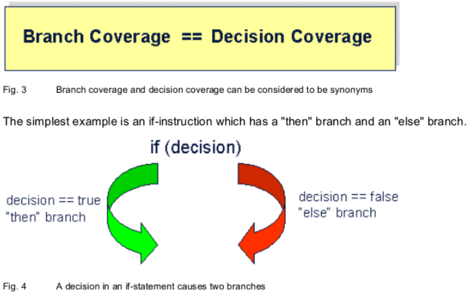
```
√   int a = 0;
√   if (decision)
    {
√         a = 1;
    }
√   a = 1 / a;
```

Figure : Statement Coverage Example

- Here a value is zero intially and if decision = 1 a value is intialized to 1.

- but when decision is 0 a value will be zero only, then a division by zero would have been occurred in the last statement
- As seen in the above example bugs will be left out even after we have achieved 100 percentage statement coverage
- 100 percentage statement coverage doesnot give 100 percentage quality

Branch Coverage

- Every point of entry and exit in the program has been invoked at least once, and
- every decision in the program has taken on all possible outcomes at least once.
- 

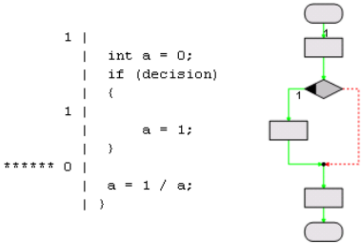
Branch Coverage == Decision Coverage

Fig. 3 Branch coverage and decision coverage can be considered to be synonyms

The simplest example is an if-instruction which has a "then" branch and an "else" branch.

Fig. 4 A decision in an if-statement causes two branches

Figure : Statement Coverage Example

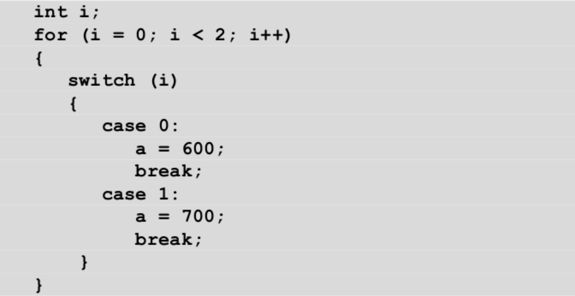


```
     1 |
       |  int a = 0;
       |  if (decision)
       |  {
     1 |
       |      a = 1;
       |  }
****** 0 |
       |  a = 1 / a;
       | }
```

Fig. 5 The invisible else-branch is taken into account for branch coverage

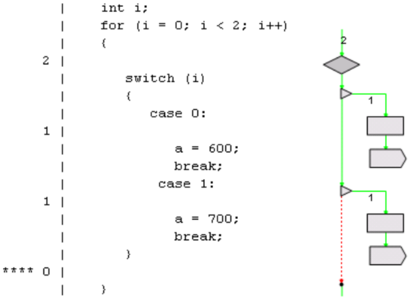
Figure : Branch coverage Example

- The branch with zero executions is marked by asterisks.
- Executed branches are in green. The branch coverage for the code excerpt is 50 percentage
- This makes branch coverage a more valuable measure than statement coverage.



```
int i;
for (i = 0; i < 2; i++)
{
  switch (i)
  {
    case 0:
      a = 600;
      break;
    case 1:
      a = 700;
      break;
   }
}
```

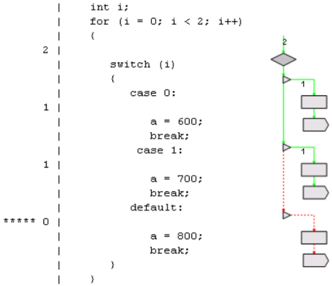
Figure : Brach Coverage Example



```
       |     int i;
       |     for (i = 0; i < 2; i++)
       |     {
     2 |
       |        switch (i)
       |        {
       |            case 0:
     1 |
       |                a = 600;
       |                break;
       |             case 1:
     1 |
       |                a = 700;
       |                break;
       |        }
**** 0 |
       |     }
```

Figure : Brach Coverage Example

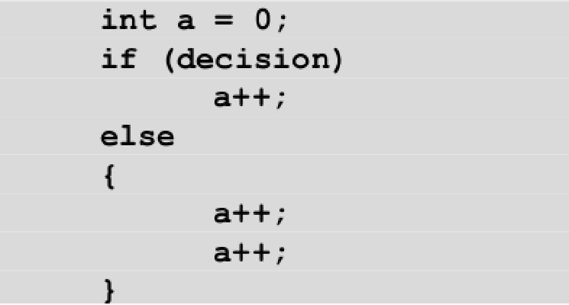
In this example we cannot achieve 100% branch coverage. Why?



```
        |     int i;
        |     for (i = 0; i < 2; i++)
        |     {
      2 |
        |        switch (i)
        |        {
        |           case 0:
      1 |
        |              a = 600;
        |              break;
        |            case 1:
      1 |
        |              a = 700;
        |              break;
        |           default:
***** 0 |
        |              a = 800;
        |              break;
        |        }
        |     }
```

Figure : With Default

Default block is hidden, even if it is not implemented explicitly.



```
int a = 0;
if (decision)
        a++;
else
{
        a++;
        a++;
}
```

Figure : Statement-Branch

- 100 % Branch coverage will Cover 100 % statements in the code
- But we cannot say that 50 % of branch coverage is covering 50 % of statements in the code.
- In the above code decision ==1 test case wil cover less number of statements than the decision == 0. But when we execute any one of the test case we can say that 50 % of branch coverage is achieved.
- For the decision == 1 we have 50 % of Branch coverage and 25 % statement coverage.

References I

[1] Torbjrn Ryber *"ESSENTIAL SOFTWARE TEST DESIGN "*, Chapter 4 Chapter 5

Thank you