Introduction

Main goal of software Testing is to make sure system will be error free and fulfill customer requirements. To handle these problems, there are two testing techniques:

• Black Box Testing

That checks the program meets the customer’s requirements.

• White Box Testing

That ensure the program is generating the accurate results or not.

Data-flow testing is also a white box testing technique which is used to detect improper use of data values due to coding errors.

Errors are not predictable, for example:

int y; if (y ==10) {};

That is allowed in programming, but it generates error.

To solve this type of error using data flow testing, we use following steps:

¬ Model program as control flow graph.

¬ Association is established between the definitions and uses of variables.

¬ Test suit is created using paths in previous steps.

Data Flow Testing

Data flow testing technique only checks a part of data. In this technique we examine the patterns and constructs the control flow graph of the program code. In this graph, nodes represent the processing statements (definitions, calculations) while edges represent the flow of control between processing statements. This testing technique is better than before testing techniques because it generates results in richer test suite.

We use notations for patterns, with pre-defined function. We use example of an application to calculate the bill of a cellular service customer depending upon on his/her usage, to define pattern’s notations. ¬d – defined, created, initialized

¬k – killed

¬u – used

¬~x - indicates all prior actions are not of interest to x

¬x~ - indicates all post actions are not of interest to x

Types

This testing is performed in two ways:

¬ Static data-flow testing:

In static testing the data is analyzed without executing it, but it is not always enough because Static Data-flow testing will fail in situations where the state of a data variable cannot be determined by just analyzing the code Testing for this programs discovered the following anomaly: Bill: define – define This is not enough in all cases like in case of arrays indexes are difficult to identify.

¬ Dynamic data-flow testing: In dynamic testing we find all possible bugs during the execution of program. For this purpose, we find all the definitions of variables, where they’re used and all the consequences are observed. In dynamic testing, test cases are created which trace every definition to each of its use and every use is traced to each of its definition.

Selection of test cases

For selection of test cases, we need to analyze the relative strength of the dataflow testing strategies. Figure depicts the relative strength of the data-flow strategies. According to the figure, the strength of testing strategies reduces along the direction of the arrow. Hence ALL PATHS is the strongest testing strategy. Also note that ACU+P and APU+C run parallel hence they are comparable

Deliberation

Dynamic data flow testing in previous testing provides strongest testing strategy by providing paths.These paths helps to create test cases.

Conclusion and Reference

That gives the concept of data flow testing and how it is use to find error and design test cases. All this work done on test example of mobile billing program.