Chapter 2
Structural Coverage Analysis Methods

Parnasi Retasbhai Patel
Charotar University of Science and Technology, India

Chintan Bhatt
Charotar University of Science and Technology, India

ABSTRACT

Structural coverage analysis for any code is a very common approach to measure the quality of any test suit. Structural coverage determines which structure of the software or which portion is not exercised. This chapter describes two different phases to achieve structural coverage analysis using DO-178B/C standards. Statement coverage is the very basic coverage criteria which involves execution of all the executable statements in the source code at least once. Analysis of structural coverage can be done by capturing the amount of code that is covered by the airborne software. The first phase contains the instrumentation procedure which instruments the source code at execution time, and the second phase is generating a report that specifies which portion of source code is executed and which one is not in the form of a percentage.

1. INTRODUCTION

Software is the crucial thing which is used everywhere in our society from education to medical and business. Some of the devices are operatable on basis of softwares. In order to increase the quality, software testing is important. Testing is one of the most important phase of software life cycle. Software testing is a process of finding errors to reduce the damage and improve the reliability, dependency and quality
Structural Coverage Analysis Methods

of software (Zander, Schieferdecker, & Mosterman, 2011). Different standards are provided to check the criticality of software.

From the last few decades, control and coverage have been employed in most of functional areas of aerospace industries including navigation, flight control and other avionics softwares. Testing of developed code is core part of software development to protect civil aviation safety and reliability of software for airborne equipments. It is most important to check the correctness and accuracy of code for any software to avoid software failures and accidents. Each requirement contains one or more test-cases to prove its correctness.

DO-178B/C and DO-278 are used to assure safety of avionics software and control systems. DO-178B/C provides guidance and standards for software development, verification, configuration management and the interface to approval authorities. It prescribes a procedure which is being followed in the software development of airborne systems by developers. Based on a safety criteria DO-178B provides different levels which are shown in Table 1.

2. TESTING TECHNIQUES

Testing involves mainly two techniques such as black box testing and white box testing.

2.1 Black Box Testing

In this type of testing only functional specifications are required. Internal software structure is not required in this kind of software testing strategy.

Table 1. Levels according to failure conditions

<table>
<thead>
<tr>
<th>Level</th>
<th>Failure Conditions</th>
<th>Coverage Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Catastrophic failure condition or cascading system failure</td>
<td>MC/DC, Branch &amp; Statement Coverage</td>
</tr>
<tr>
<td>B</td>
<td>Hazardous failure condition</td>
<td>Branch &amp; Statement Coverage</td>
</tr>
<tr>
<td>C</td>
<td>Major system failure</td>
<td>Statement Coverage</td>
</tr>
<tr>
<td>D</td>
<td>Minor system failure</td>
<td>None</td>
</tr>
<tr>
<td>E</td>
<td>No effect on system</td>
<td>None</td>
</tr>
</tbody>
</table>
Related Content

Structural Coverage Analysis Methods
www.igi-global.com/chapter/structural-coverage-analysis-methods/219975?camid=4v1

Evaluating the Refactoring Index Using Entropy Approach
www.igi-global.com/chapter/evaluating-the-refactoring-index-using-entropy-approach/219982?camid=4v1a

Teaching Model-Driven Engineering in a Master’s Program: Three Editions on a PBL-Based Experience
www.igi-global.com/chapter/teaching-model-driven-engineering-in-a-masters-program/219980?camid=4v1a
Operator Overloading as a DSL Parsing Mechanism
www.igi-global.com/chapter/operator-overloading-as-a-dsl-parsing-mechanism/219983?camid=4v1a