Call Coverage with Testwell CTC++ and Imagix 4D

What is Call Coverage?

While function coverage reflects if every function in a program was called during test at all, for call coverage all *possible* function calls are under consideration.

If every possible call was executed under test, full call coverage is achieved. For every single function, the percentage of call coverage can be measured as ratio of executed calls to possible calls.

The safety standard ISO 26262-6 recommends measuring call coverage as one of two alternative methods for integration tests (Section 10.4.5, the alternative is function coverage).

Call coverage is also known as call pair coverage.

Deduction with Testwell CTC++

The code coverage analyzer Testwell CTC++ measures function coverage directly. Call coverage is not measured directly, but can be derived from stronger coverage measures[1]:

If 100% condition coverage can be achieved for the whole software under test, then all function calls were executed. So full call coverage is achieved.

For this conclusion, it is only necessary to measure condition coverage if function calls appear in composite conditions like

```
if (x == 5 && foo())
```

In this example, the condition `x == 5` must be evaluated as true, so that foo is called at this place.

If these kinds of calls do not occur, 100% statement coverage is sufficient to prove call coverage.

Detailed Analysis with Imagix 4D

By using the graphical analysis tool Imagix 4D, the evaluation of call coverage can be further refined. The basis is a decision coverage report generated with Testwell CTC++ using the xml_imagix4d template. For call coverage, Imagix 4D provides a dedicated report that displays both the decision coverage and the call coverage as a percentage for each function. Additionally, the total number of function calls is listed.

[1] This derivation is not valid in a strict logical sense, if function pointers or comparable constructs in C++ are used. However, in the context of ISO 26262-6 requirements, a higher coverage level should still be enough to replace one of the weaker measures function or call coverage.

The report then lists all missed calls. Optionally, the actually executed calls (“taken”) can also be displayed. The last two columns contain information about the file and line number where the respective function or function call is located.

File Callee Caller	Decision Coverage	Call Coverage	Total Calls Covered?	File	Line
myhome.c					
main	57%	-	0	myhome.c	10
Function has no callers					
regulators.c					
air_condition	0%	0%	1	regulators.c	42
temperature_control			missed	regulators.c	69
close_windows	100%	50%	2	regulators.c	20
main			missed	myhome.c	28
open_windows_for			taken	regulators.c	34
heat	100%	50%	2	regulators.c	37
main			missed	myhome.c	31
temperature_control			taken	regulators.c	58
lights	75%	67%	3	regulators.c	4
main			missed	myhome.c	27
main			taken	myhome.c	36
main			taken	myhome.c	60

This extended analysis is especially interesting in two cases:

- At integration test level, achieving full statement coverage or condition coverage is sometimes quite hard. In combination with Imagix 4D and Testwell CTC++, this additional testing effort can be avoided by measuring call coverage directly.
- As long as call coverage is not fully achieved, the integrated report directly shows the reasons – missing function calls are displayed and can be traced back to the source code.
  The percentage of call coverage offers a quantitative impression how far the tests are away from achieving full call coverage.