MC/DC (Modified Condition/Decision Coverage)

Por Luis Miguel García Marín

Este documento pretende responder a las siguientes preguntas sobre el criterio de cobertura MC/DC:

1. ¿Por qué crees que es importante este criterio para ser aplicado en pruebas de caja blanca?
2. Define brevemente en qué consiste el criterio usando tus propias palabras.
3. Da un ejemplo de aplicación

Respuestas:

1. ¿Por qué crees que es importante este criterio para ser aplicado en pruebas de caja blanca?

Es importante para el testeo de sistemas críticos, en los que sería conveniente probar todas las combinaciones de booleanos de condiciones que influyen en las salidas de decisiones de los programas. Nos conviene en estos casos, ya que la alternativa a probar las únicas combinaciones de condiciones que influyen en las salidas de las decisiones es probar todas las combinaciones de condiciones posibles, y eso es MCC (Multiple Condition Coverage), el cual es mucho menos viable. La explicación a esto es que con MCC tendríamos 2^C posibles combinaciones a probar (donde C es el número de condiciones por decisión), mientras que con MC/DC tendríamos C + 1 posibles combinaciones a probar. Por lo que, por ejemplo, si tuviésemos por decisión unas 10 condiciones, con MCC tendríamos que hacer 2^{10} = 1024 tests, mientras que con MC/DC tendríamos que hacer 10 + 1 = 11 tests. Es claro que con MC/DC nos ahorramos mucho más trabajo.

Además, a la hora de probar sistemas críticos como software para aviones de pasajeros, si nuestro programa falla para alguna determinada combinación de condición de entrada, las pérdidas pueden ser desastrosas (no sólo dinero, sino vidas están en juego). Si bien un 100% en los criterios de cobertura no nos garantiza un programa libre de errores, sí que nos orienta por un muy buen camino a la hora de realizar tests de caja blanca, y por esto es muy conveniente probar todas esas posibles combinaciones de condiciones de entrada, pero no absolutamente todas (ya que puede ser inasumible), si no aquellas combinaciones que realmente repercutan en la salida de las decisiones del programa.

Así por ejemplo, MC/DC se utiliza en los estándares DO-178B y DO-178C (Consideraciones de software en la certificación de equipos y sistemas aerotransportados, Comisión técnica de radio para la aeronáutica, DO-178B a la fecha de 1 de diciembre de 1992 y DO-178C a la fecha de enero de 2012), realizados y certificados por la Administración Federal de Aviación (FAA), para garantizar que el software aeroespacial DAL A se prueba correctamente.

2. Define brevemente en qué consiste el criterio usando tus propias palabras.

Este criterio de cobertura verifica que cada posible valor de una condición determina la salida de una decisión al menos una vez. Es decir, que se ha comprobado que cada una de las

condiciones afecta independientemente a la salida de la condición (se va por una rama o por otra).

3. Da un ejemplo de aplicación.

Viendo ahora un ejemplo más concreto, imaginemos que estamos desarrollando un programa que nos ha encargado el gobierno para controlar un sistema de captura de fotos para el control de tráfico de un cruce. Se nos indica que nuestro sistema debe tomar una foto si el semáforo del cruce está en rojo (ROJO) y se detecta que las ruedas delanteras de un coche están sobre la línea marcando el comienzo del cruce (LINEA), o bien, si el radar del cruce detecta que un coche está acelerando (RADAR) y se detecta que las ruedas delanteras de un coche están sobre la línea marcando el comienzo del cruce (LINEA). Queremos asegurarnos de que nuestro programa cumple el criterio de cobertura MC/DC.

- Dadas estas especificaciones, en nuestro programa llegaríamos a tener una decisión como:

```
if (LINEA && (ROJO || RADAR)) {
   tomarFoto();
} else {
   noTomarFoto();
}
```

De esta forma, los casos a probar (y por tanto, los tests que tendríamos que hacer) para cubrir el criterio de cobertura MC/DC serían:
– (LINEA=true, ROJO=true, RADAR=false) -> true (tomar foto)
– (LINEA=true, ROJO=false, RADAR=true) -> true (tomar foto)
– (LINEA=false, ROJO=true, RADAR=true) -> false (no tomar foto)
– (LINEA=true, ROJO=false, RADAR=false) -> false (no tomar foto)

Referencias adicionales a las del campus virtual:

- https://hmong.es/wiki/Code_coverage
- http://www.inf-cr.uclm.es/www/mpolo/tcymParte1.pdf
- https://es.frwiki.wiki/wiki/MC%2FDC
- https://www.ibm.com/support/pages/how-obtain-100-coverage-mcdc-test-realtime-code-coverage
- http://staff.cs.upt.ro/~marius/curs/vvs/lect3.pdf
- https://www.cse.chalmers.se/edu/year/2018/course/TDA567_Testing_Debugging_Verification/Lec2018/Examples1.pdf
- https://es.parasoft.com/blog/how-to-obtain-100-structural-code-coverage-of-safety-critical-systems/