Ruolo della copertura MC/DC nel testing strutturale secondo lo standard DO-178

Relatore
Prof. Alessandro Fantechi

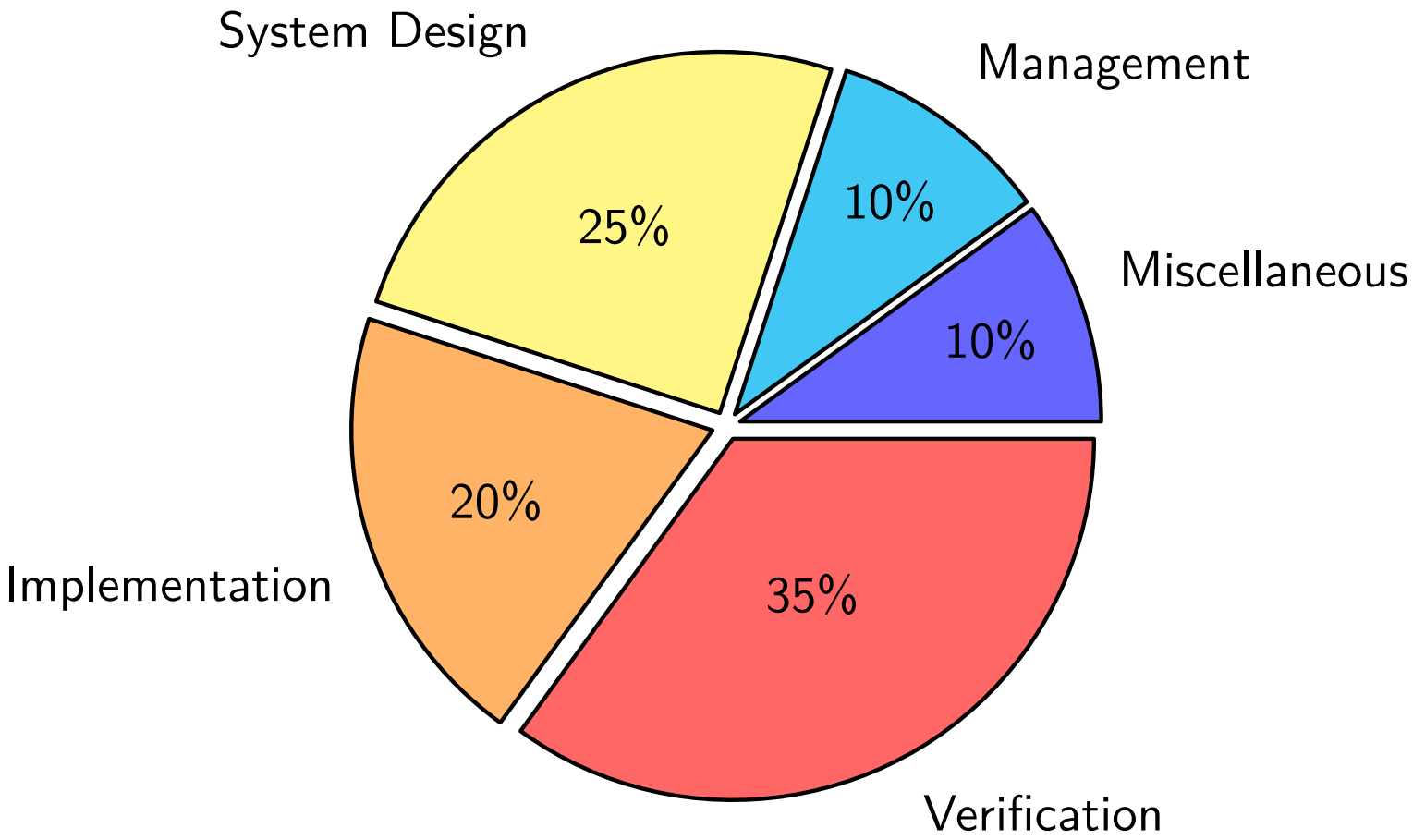
Candidato
Tommaso Gualtierotti

Cosa è lo standard DO-178

I documenti americani DO-178 e i rispettivi ED-12 europei sono una raccolta di linee guida, considerate come standard, per lo **sviluppo** e la **certificazione** del software in ambito **aerospaziale**. Gli standard sono sviluppati in collaborazione tra RTCA (Radio Technical Commission for Aeronautics) e EUROCAE (European Organisation for Civil Aviation Equipment). Questi documenti sono il punto di riferimento per la certificazione di aeromobili da parte di FAA e EASA e hanno un ruolo centrale nello svilluppo di nuove metodologie di software development e critical software testing.
Di questo standard sono state rilasciate 4 versioni dal 1981 ad oggi:

- DO-178 (1981)
- DO-178A (1985)
- DO-178B (1992)
- DO-178C (2011)

Costo del processo di sviluppo



Costo del processo di sviluppo

Livelli di Software definiti dallo standard DO-178

Livello	Gravità	Descrizione	Tasso Fallimento
A	Catastrofico	Guasti che possono creare un disastro aereo e la potenziale morte di tutti le persone a bordo.	10^{-9}
B	Critico	Guasti che possono causare una riduzione delle performance o della sicurezza dell'aeromobile. Guasti che possono portare alla morte o al grave ferimento delle persone a bordo.	10^{-7}
C	Maggiore	Guasti che possono ridurre performance e sicurezza dell'aeromobile o un significativo aumento del carico di lavoro richiesto ai piloti. Guasti che possono portare al ferimento di persone a bordo	10^{-5}
D	Marginale	Guasti che hanno un impatto sul carico di lavoro richiesto ai piloti. Guasti che possono provocare disagi alle persone a bordo.	10^{-3}
E	Trascurabile	Guasti che non hanno nessuna ricaduta su passeggeri e equipaggio.	Non definito

Livelli software definiti dallo standard DO-178

Criterio di Copertura	Livello Software
Statement Coverage	A, B, C
Decision Coverage	A, B
MC/DC	A

Requisiti di copertura strutturale del testing richiesti dal DO-178

MCC

↓

MC/DC

↓

Condtion/Decision Coverage

↓

Decision Coverage

↓

Statement Coverage

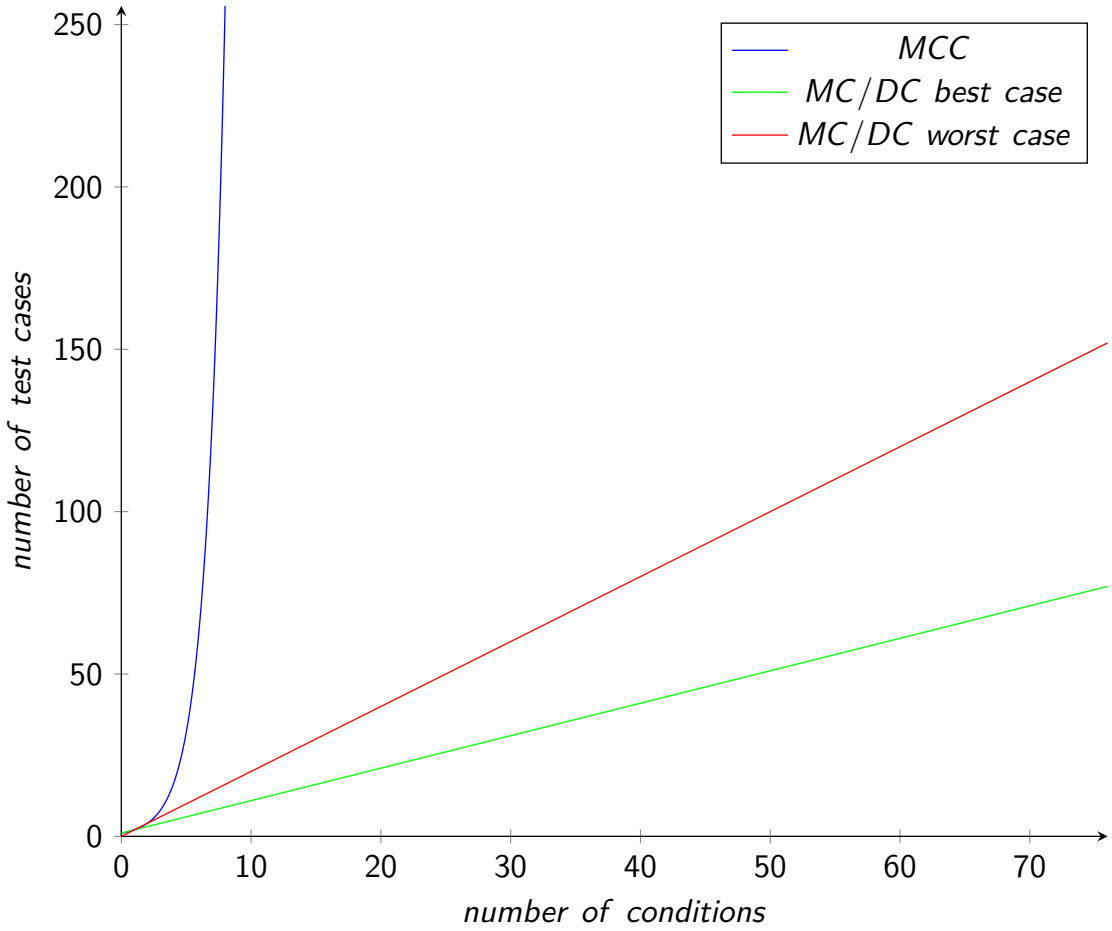
Gerarchia di Sussunzione per misure di copertura sul controllo del flusso

Definizione di MC/DC

1. Ogni punto di entrata e uscita nel programma è stato invocato almeno una volta,
2. ogni condizione in una decisione nel programma deve aver assunto tutti i possibili valori almeno una volta,
3. ogni decisione nel programma ha assunto tutti i possibili valori almeno una volta,
4. ogni condizione è stata mostrata influenzare in modo indipendente il risultato della decisione della quale fa parte:
    1. variando il valore della condizione in esame mentre vengono fissati i valori di tutte le altre condizioni della decisione.
    2. variando il valore della condizione in esame mentre vengono fissati i valori delle **sole** condizioni che possono influenzare il risultato della decisione.

Se sono soddisfatti i precedenti requisiti si può notare che sono necessari **N + 1** casi di test, dove **N** è il **numero di condizioni** all'interno della decisione presa in esame.
In caso di **condizioni fortemente accoppiate**, ovvero nel caso in cui la stessa condizione appaia più volte nella stessa decisione, risulta molto **più difficile trovare una suite di test** che rispetti i requisiti sopra elencati.
Per fare ciò quindi esistono diverse interpretazioni di copertura MC/DC e potenzialmente i casi di test possono aumentare fino ad un totale massimo di **2N** nel **caso peggiore**.

	Numero di condizioni, n									
	1	2	3	4	5	6-10	11-15	16-20	21-35	36-76
numero di espressioni booleane con n condizioni	16491	2262	685	391	131	219	35	36	4	2



Definizione dell'esperimento

L'esperimento che viene condotto qui di seguito, si basa su una funzione scritta in linguaggio C, che simula le varie modalità di un cruise control. L'esperimento viene condotto su tre versioni differenti della funzione, denominate A, B e C.

Come specificato in "Hayhurst, Kelly J. 2001 A Practical Tutorial on Modified Condition/Decision Coverage NASA/TM-2001-210876", uno dei temi che sono più al centro di discussione è il fatto che, una decisione è una qualsiasi espressione Booleana, e perciò la copertura MC/DC si applica a tutte le decisioni presenti all'interno del codice sotto test, e non solo alle espressioni interne ad un costrutto condizionale.

L'obiettivo è quindi valutare le differenze tra gli strumenti presi in considerazione soprattutto capire il modo nel quale valutino le decisioni.

Cosa è una decisione

Consideriamo le seguenti due espressioni booleane:

$$A = B \text{ or } C \qquad \text{Espressione 1}$$

$$E = A \text{ and } D \qquad \text{Espressione 2}$$

Queste due espressioni sono logicamente equivalenti a:

$$E = (B \text{ or } C) \text{ and } D \qquad \text{Espressione 3}$$

- Tutte e tre le precedenti espressioni sono decisioni, anche se nessuna delle espressioni si trova all'interno di un costrutto condizionale (cfr. if, for, while).
- Decisione non è sinonimo di ramificazione, perciò la copertura MC/DC deve essere calcolata su tutte e tre le espressioni.
- Sebbene la terza decisione sia equivalente alle prime due, non è detto che una suite di test che raggiunga il 100% di copertura MC/DC sulle prime due decisioni raggiunga il 100% anche sulla terza decisione.

Elenco tools presi in considerazione

- **RapiCover** e **CANTATA**
  - Licenza: Prova Gratuita disponibile previa Videoconferenza
  - Non selezionato
- **Reactis**
  - Licenza: Prova Gratuita di 30 giorni
  - Selezionato
- **VectorCast**
  - Licenza: Prova Gratuita
  - Non selezionato
- **Testwell CTC++**
  - Licenza: Prova Gratuita di 30 giorni
  - Selezionato
- **PARASOFT C/C++ TEST**
  - Licenza: Prova Gratuita
  - Non selezionato

Descrizione tools utilizzati per l'esperimento

Reactis for C

- Composto di tre strumenti principali: tester, simulatore e validatore.
- Si individua la funzione di ingresso, ovvero la funzione che viene eseguita non appena entra in funzione il simulatore, e si definisce una lista di parametri di input e di output.
- Si carica la suite di test e si esegue il programma, mostrando un report finale che evidenzia i risultati di copertura sui principali criteri.

Testwell CTC++

- Composoto da due strumenti: preprocessore, postprocessore.
- Il preprocessore è responsabile di compilare i vari file sorgente e linkarli in un eseguibile che verrà poi eseguito e sul quale verranno condotti i test.
- Il programma viene eseguito e finita l'esecuzione del programma, Testwell CTC++ genera dei file che contengono informazioni riguardo all'esecuzione.
- Il postprocessore viene istruito sul criterio di copertura che vogliamo visualizzare nel report di esecuzione.

Versione A

```
1   int g_dsMode = M_NOTINIT;
2
3   int Mode (int deactivate, int activate, int onOff, int set)
4   {
5       if ( g_dsMode == M_OFF && onOff )
6           g_dsMode = M_INIT;
7       else if ( g_dsMode == M_NOTINIT || !onOff )
8           g_dsMode = M_OFF;
9       else if ( g_dsMode == M_INIT && (set && !deactivate) )
10          g_dsMode = M_ACTIVE;
11      else if ( g_dsMode == M_INACTIVE && activate )
12          g_dsMode = M_ACTIVE;
13      else if ( g_dsMode == M_ACTIVE && deactivate )
14          g_dsMode = M_INACTIVE;
15
16      return g_dsMode;
17  }
```

Versione B

```
1  int g_dsMode = M_NOTINIT;
2
3  int Mode (int deactivate, int activate, int onOff, int set)
4  {
5      if ( g_dsMode == M_OFF && onOff )
6          g_dsMode = M_INIT;
7
8      else if ( g_dsMode == M_NOTINIT || !onOff )
9          g_dsMode = M_OFF;
10
11     else if ( (g_dsMode == M_INIT && (set && !deactivate)) ||
12               ( g_dsMode == M_INACTIVE && activate ) )
13
14         g_dsMode = M_ACTIVE;
15
16     else if ( g_dsMode == M_ACTIVE && deactivate )
17         g_dsMode = M_INACTIVE;
18
19
20     return g_dsMode;
21 }
```

Versione C

```
1  int g_dsMode = M_NOTINIT;
2
3  int Mode(int deactivate, int activate, int onOff, int set)
4  {
5
6      int caseA = (g_dsMode == M_NOTINIT !onOff );
7      int caseB = (g_dsMode == M_OFF && onOff );
8      int caseC = (g_dsMode == M_INIT && (set && !deactivate) );
9      int caseD = (g_dsMode == M_INACTIVE && activate);
10     int caseE = (g_dsMode == M_ACTIVE && deactivate);
11
12     if (caseA)
13         g_dsMode = M_OFF;
14
15     else if (caseB)
16         g_dsMode = M_INIT;
17
18     else if (caseC || caseD)
19         g_dsMode = M_ACTIVE;
20
21     else if (caseE)
22         g_dsMode = M_INACTIVE;
23
24     return g_dsMode;
25 }
```

Caso	deactivate	activate	onOff	set
1	0	1	1	1
2	1	0	0	0
3	0	0	1	0
4	0	0	1	0
5	1	0	1	1
6	0	0	1	1
7	0	0	1	0
8	1	0	1	0
9	0	0	1	0
10	0	1	1	0

Suite di test T_A

Caso	deactivate	activate	onOff	set
1	0	1	1	1
2	0	0	1	0
3	0	0	1	1
4	1	0	1	0
5	0	0	1	0
6	0	1	1	0

Suite di test T_C

Confronto Risultati

Testwell CTC++ outcomes

	DC	MC/DC	MCC
T_A	12/12	23/23	28/28
T_C	12/12	18/23	23/28

Versione A

	DC	MC/DC	MCC
T_A	10/10	21/21	25/29
T_C	10/10	16/21	20/29

Versione B

	DC	MC/DC	MCC
T_A	10/10	33/33	39/39
T_C	10/10	28/33	34/39

Versione C

Reactis outcomes

	DC	MC/DC	MCC
T_A	10/10	11/11	16/16
T_C	10/10	6/11	11/16

Versione A

	DC	MC/DC	MCC
T_A	8/8	11/11	15/19
T_C	8/8	6/11	10/19

Versione B

	DC	MC/DC	MCC
T_A	8/8	5/5	9/9
T_C	8/8	5/5	9/9

Versione C

Legenda: 100% invariato | 100% diminuito | <100% diminuito

Analisi dei Risultati

- La Versione A e B della funzione danno gli stessi risultati di copertura su entrambi gli strumenti.
- La Versione C presenta delle differenze:
  - **T_A** Si ottiene il 100% di copertura con entrambi i tools ma Reactis si comporta in modo errato.
  - **T_C** Reactis riesce ad ottenere il 100% di copertura MC/DC con la suite T_C mentre Testwell CTC++ non riesce, poiché Reactis non riconosce le espressioni booleane di assegnamento come decisioni (come con T_A).

Questo dimostra la **necessità** e **l'importanza** che uno **strumento** utilizzato come supporto durante il ciclo di vita dello sviluppo di software safety critical **debba essere qualificato**, infatti mentre **Testwell CTC++** è **qualificato** secondo lo standard DO-178, **Reactis non lo è**.

Conclusioni e Sviluppi Futuri

Concludendo quindi si possono **proporre** ulteriori **ampliamenti del lavoro condotto** che seguono:

- Condurre esperimenti più estesi su codice provienente da diversi progetti.
- Effettuare una comparazione che comprenda un maggior numero di tools di supporto al testing, comparandoli su un maggior numero di caratteristiche (ricordiamo infatti che il poco tempo a disposizione non ha permesso di richiedere le prove degli strumenti in anticipo, o di aspettare le risposte da parte delle aziende.
- Proprorre una collaborazione a un'azienda che utilizzi uno degli strumenti citati in precedenza ed effettuare quindi un esperimento che occupi un tempo maggiore senza avere limitazioni sul periodo di prova del software.