RAPITA
SYSTEMS LTD

Modified Condition/Decision Coverage (MC/DC) of Ada Case Statements

Ada Europe 2014

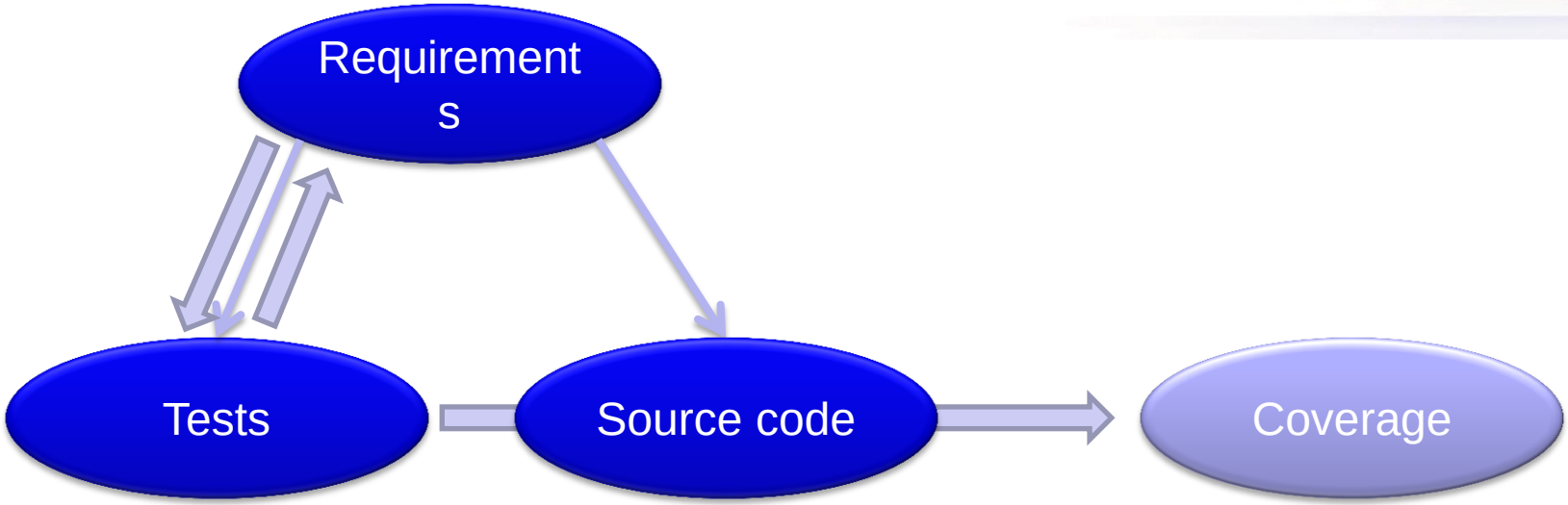
Andrew Coombes, Roger Braff, (Rockwell Collins) Jack Whitham, Antoine Colin

acoombes@rapitasystems.com

Context

```
case x is
  when 1 => do_alpha;
  when 2|3 => do_beta;
  when 5..6 => do_gamma;
end case;
```

Context: Why structural coverage?



Checks:

- Does every requirement have a test? [traceability]
- Is every test associated with a requirement? [traceability]
- Do the tests exercise all of the source code? [coverage]
  - No – code is unnecessary
  - No – requirements are missing
  - No – tests are not detailed enough

Introduction to MC/DC

Structural code coverage technique

MC/DC = Modified Decision/Condition Coverage

- What's a condition?
  - *A Boolean expression containing no Boolean operators*
  - For example:
    - (a > 17)
    - Weight_on_wheels
- What's a decision?
  - *Boolean expression composed of conditions and zero or more Boolean operators.*
  - For example:
    - `if (a > 17) and not Weight_on_wheels then …`
  - Decision includes:
    - Branch points
    - Boolean operations that appear on assignment statements
    - Actual parameters
    - Etc.

Introduction to MC/DC

Defined in DO-178B as:

- *Every point of entry and exit in the program has been invoked at least once,*
- *every condition in a decision in the program has taken all possible outcomes at least once,*
- *every decision in the program has taken all possible outcomes at least once,*
- *and each condition in a decision has been shown to independently affect that decision's outcome.*
- *A condition is shown to independently affect a decision's outcome by varying just that condition while holding fixed all other possible conditions.*

```
begin
   if True or True or True or True then
      isAdaEurope := False;
   elsif place = Paris then
      isAdaEurope := True;
   else
      isAdaEurope := False;
   end if;
end;
```

Why when is a decision?

Are the when statements decisions?

Why?

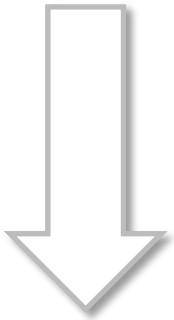
No:

- Case labels aren't boolean expressions. Therefore, can't be decisions

Yes:

- If statements are morphologically equivalent to case statements
- If not, it allows a "back-door" to avoid MC/DC testing obligations

```
case x is
     when 1 => … ;
     when 2 => … ;
     when others => …;
end case;
```



```
if x = 1 then
     …;
elsif x = 2 then
     …;
else
     …;
end if
```

If when isn't a decision

When statements

```
17 40      case a is
18            when 0 =>
19 ...        result := 0;
20            when 1 | 2 | 3 =>
21 ...        result := 1;
22            when 4 .. 6 =>
23 ...        result := 2;
24            when others =>
25 ...        result := 3;
26
27         end case;
```

Test cases (4)

```
testdriver(testcase (0),0);
testdriver(testcase (1),1);
--  a :=  testcase (2);
--  a :=  testcase (3);
--  a :=  testcase (4);
testdriver(testcase (5),2);
testdriver(testcase (11),3);
```

Statement Coverage (6/6 100.000%)

Statement				
Name	Location	C-Stmt	#TestCases	Justifications
S34	main_ada.adb:17	✔ T	1	-
S35	main_ada.adb:19	✔ T	1	-
S36	main_ada.adb:21	✔ T	1	-
S37	main_ada.adb:23	✔ T	1	-
S38	main_ada.adb:25	✔ T	1	-
S39	main_ada.adb:28	✔ T	1	-

100% statement coverage

If when isn't a decision

if statements

```
34  ...      if a = 0 then
35  29          result := 0;
36  30       elsif a = 1 or a = 2 or a = 3 then
37  34          result := 1;
38  35       elsif a >= 4 and a <= 6 then
39  37          result := 2;
40           else
41  38          result := 3;
42           end if;
```

Test cases (7)

```
testdriver(testif (0),0);
testdriver(testif (1),1);
testdriver(testif (2),1);
testdriver(testif (3),1);
testdriver(testif (-10),3);
testdriver(testif (4), 2);
testdriver(testif (11),3);
```

MC/DC Decision			Conditions
Name	Location	C-MC/DC-Decision	C-MC/DC-Condition*
MCDC127	main_ada.adb:34	✔ T	1 / 1
C1: a = 0	main_ada.adb:34	- ✔	T
MCDC136	main_ada.adb:36	✔ T	3 / 3
C1: a = 1	main_ada.adb:36	- ✔	T
C2: a = 2	main_ada.adb:36	- ✔	T
C3: a = 3	main_ada.adb:36	- ✔	T
MCDC145	main_ada.adb:38	✔ T	2 / 2
C1: a >= 4	main_ada.adb:38	- ✔	T
C2: a <= 6	main_ada.adb:38	- ✔	T

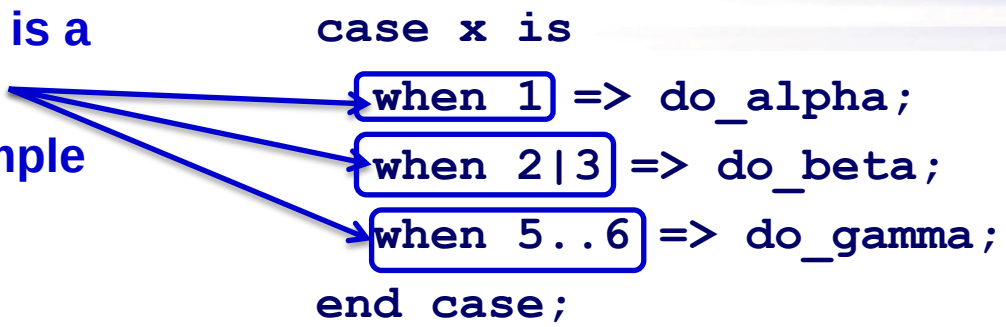
100% MC/DC

How when is handled as a decision

- Each when statement is a separate decision
- A single value is a simple comparison

  x = 1
- A choice is a disjunction between conditions

  x = 2 or x = 3
- A range has a number of possible implementations…



How when is handled as a decision

Implementing ranges

```
case x is
  when 12..18 => ...
```

Option 1: use "in"

```
if x in 12..18 then ...
```

- Requires 2 tests: x not in range, x in range
- Might not be acceptable to certifying authorities

Option 2: use pairs of tests

```
if x >= 12 and x <= 18 then ...
```

- Requires 3 tests: x < 12, 12 >= x >= 18, x > 18
- Problematic in some situations…

Option 3: use individual tests

```
if x = 12 or x = 13 or ... or x = 18 then
  ...
```

- Requires 8 tests: x = 12, x = 13, … x = 18, x = 1
- Number of tests scales with the size of the range

When when causes a problem

Implementing MC/DC checks on ranges using pairs of tests

Problem 1:

- One of the tests is at the limit of the type size

```
x : natural;
case x is
  when 0..5 => ... - can't create a test case where x < 0
```

Problem 2:

- Contiguous ranges of numbers

```
case x is
  when 0..12 => ...
  when 13..17 => ...
```

- Equivalent to:

```
if x >= 0 and x <= 12 then ...
elsif x >= 13 and x <= 17 then ...
```

- Can't create a test case in the elsif branch where `x >= 13` is false
  - This condition is dead code

Summary

Consensus seems to be "when" is a decision

Implications for code coverage tool vendors…

Current status at Rapita

- Working implementation of when coverage within RapiCover
  - In a development branch of the tool
- Will be released at the next RVS release

Thank-you for your attention

For further details:

acoombes@rapitasystems.com