Modified Condition/Decision Coverage (MC/DC) Tutorial

Leanna Rierson (FAA)
Phone: 202.267.3785
E-mail: Leanna.Rierson@faa.gov

Kelly Hayhurst (NASA Langley Research Center)
Phone: 757.864.6215
E-mail: k.j.hayhurst@larc.nasa.gov

Dan Veerhusen (Rockwell Collins, Inc.)
Phone: 319.295.4874
E-mail: dsveerhu@cca.rockwell.com

Objectives

- **Explain the role of MC/DC within the DO-178B verification process**
- **Describe a method for evaluating requirements-based test cases for MC/DC**
- **Explain the process for evaluating an applicant's MC/DC data**
- **Discuss common problems associated with MC/DC**

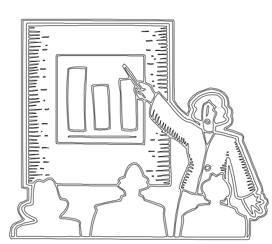
FAA National Software Conference - June 6, 2001 2

Overview

- Module 1 – Defining MC/DC
- Module 2 – An MC/DC Approach
- Module 3 – MC/DC Compliance Assessment
- Module 4 – Common Pitfalls

FAA National Software Conference - June 6, 2001 3

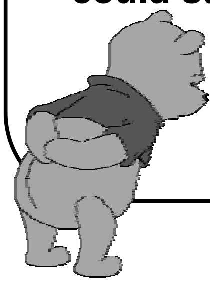
Module 1
Defining MC/DC



FAA National Software Conference - June 6, 2001 4

"Here is Edward Bear, coming downstairs now, bump, bump, bump, on the back of his head, behind Christopher Robin. It is, as far as he knows, the only way of coming downstairs, but sometimes, he feels that there is another way... if only he could stop bumping for a moment and think of it!"

- Winnie-the-Pooh, A. A. Milne



FAA National Software Conference - June 6, 2001 5

Module 1 Overview

Context for MC/DC

- Role of coverage in DO-178B life cycle processes
- Types of coverage
- MC/DC details
  - description of MC/DC in DO-178B
  - other things you need to know
  - unique cause versus masking

FAA National Software Conference - June 6, 2001 6

A Note about Notation

- Boolean operators are denoted by bolded italics: ***and***, ***or***, ***xor***, ***not***
- Boolean conditions are denoted by bolded capital letters: **A, B, C, …**
- Non-Boolean variables are denoted in plain lower case letters: x, y, z, …
- Boolean outcomes are written as either *false* or *true*, or F or T
- A test case for a Boolean function with *n* inputs is denoted by $c = (c_1 \, c_2 \ldots c_n)$, where c_i = F or T

FAA National Software Conference - June 6, 2001 7

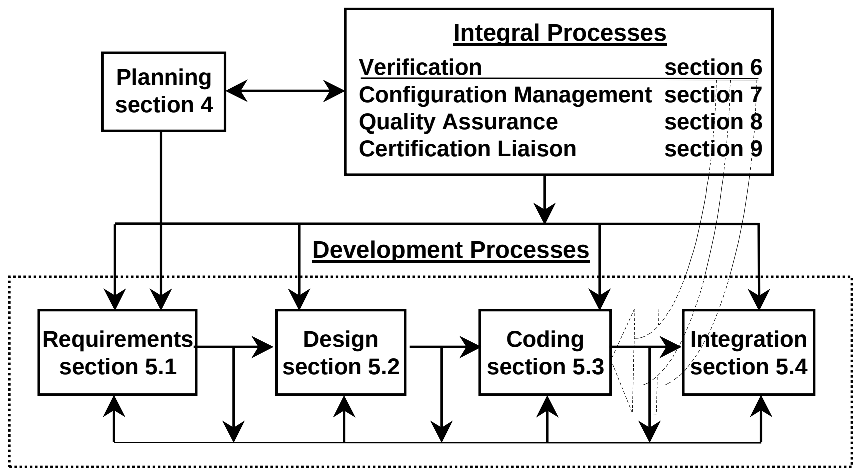
Verification Process

"No product of human intellect
comes out right the first time.
We rewrite sentences,
rip out knitting stitches,
replant gardens,
remodel houses,
and repair bridges.
Why should software be any different?"

- Lauren Wiener, Digital Woes

FAA National Software Conference - June 6, 2001 8

DO-178B Software Life Cycle Processes

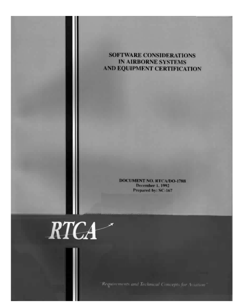


FAA National Software Conference - June 6, 2001 9

Coverage

Coverage is a measure -- not a test

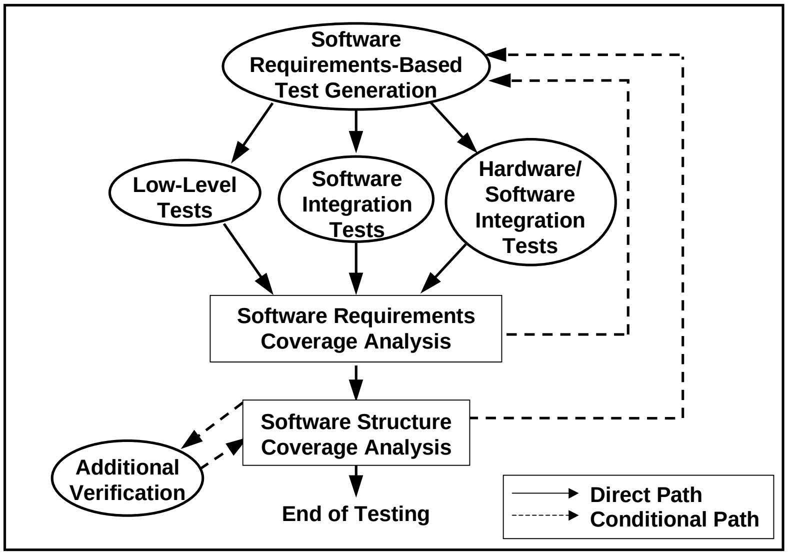
- Coverage is the extent to which a verification activity has satisfied its objectives
  - for testing, coverage can be used as an exit criteria
- DO-178B calls out 2 coverage measures
  - requirements coverage
  - software structure coverage
    - structural coverage



FAA National Software Conference - June 6, 2001 10

Leanna Rierson, Kelly Hayhurst, Dan Veerhusen

DO-178B Software Testing Activities



Requirements Coverage Analysis

Table A-7
Verification of Verification Process Results

	Objective		Applicability by SW Level			
	Description	**Ref.**	**A**	**B**	**C**	**D**
1	Test procedures are correct.	6.3.6b	○	○	○	
2	Test results are correct and discrepancies explained.	6.3.6c	○	○	○	
3	Test coverage of high-level requirements is achieved.	6.4.4.1	○	○	○	○
4	Test coverage of low-level requirements is achieved.	6.4.4.1	○	○	○	

- **These are necessary requirements -- but not sufficient**

Why is this not enough?

Intent of Structural Coverage

Structural Coverage Analysis provides a means to confirm that the requirements-based tests exercised the code structure

- **The intent of structural coverage is to:**
  - **provide evidence that the code structure was verified to the degree required for the applicable software level**
  - **provide a means to support demonstration of absence of unintended functions**
  - **establish the thoroughness of requirements-based testing**

- from FAQ #43, DO-248A

FAA National Software Conference - June 6, 2001 13

Types of Structural Coverage

- **Statement Coverage**
- **Decision Coverage**
- **Condition Coverage**
- **Condition/Decision Coverage**
- **Modified Condition/Decision Coverage**
- **Multiple Condition Coverage**

FAA National Software Conference - June 6, 2001 14

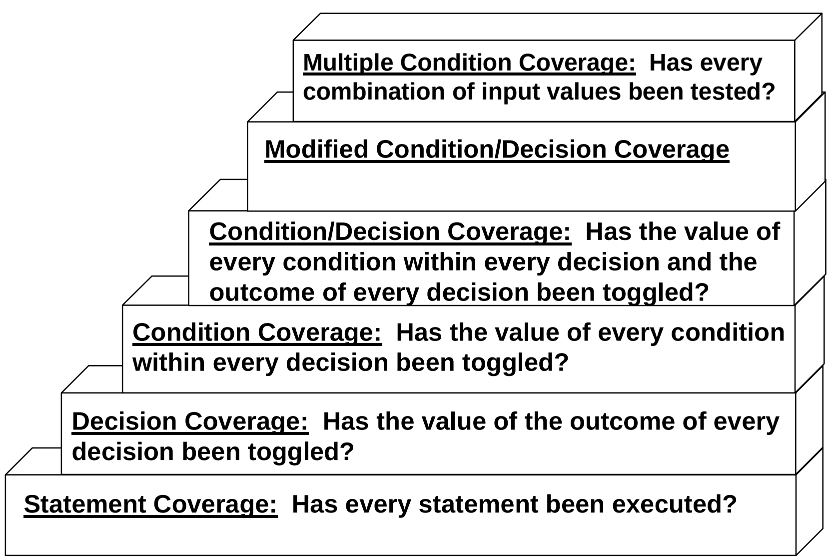
Structural Coverage in DO-178B

Table A-7
Verification of Verification Process Results

	Objective		Applicability by SW Level			
	Description	Ref.	A	B	C	D
1	Test procedures are correct.	6.3.6b	○	○	○	
5	Test coverage of software structure (modified condition/decision) is achieved.	6.4.4.2	○			
6	Test coverage of software structure (decision coverage) is achieved.	6.4.4.2a 6.4.4.2b	○	○		
7	Test coverage of software structure (statement coverage) is achieved.	6.4.4.2a 6.4.4.2b	○	○	○	

FAA National Software Conference - June 6, 2001 15

Where MC/DC Fits



FAA National Software Conference - June 6, 2001 16

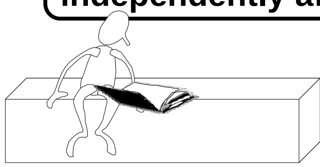
MC/DC Description

① every point of entry & exit in the program has been invoked at least once

② every condition in a decision in the program has taken all possible outcomes at least once

③ every decision in the program has taken all possible outcomes at least once

④ each condition in a decision has been shown to independently affect that decision's outcome



FAA National Software Conference - June 6, 2001 17

Conditions & Decisions

Condition:

– a Boolean expression containing no Boolean operators

– includes Boolean valued expressions with relational operators, such as, > , < , =

- for example, x > y

Decision:

– a Boolean expression composed of zero or more Boolean operators. A decision without a Boolean operator is a condition.

– if a condition appears more than once in a decision, each occurrence is a distinct condition

FAA National Software Conference - June 6, 2001 18

Hmmmm...

How many conditions are in the expression

(A *and* B) *or* (B *and* C) *or* (A *and* C)?

- A condition is said to be coupled with another condition if changing the value of the condition affects the value of the other

Is Z:= ((x > y) or B); considered a decision?

- A decision is not always at a branch point

FAA National Software Conference - June 6, 2001 19

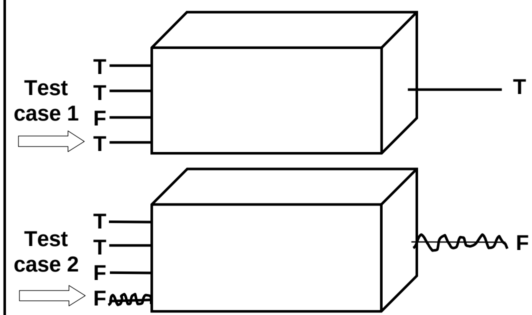
Independent Effect

- A condition independently affects a decision's outcome if that condition *alone* determines the outcome of the decision
- Two methods for showing the independent effect of a condition are:
  - unique cause
  - masking
- Unique cause may be implied by the MC/DC description in the Glossary of DO-178B

FAA National Software Conference - June 6, 2001 20

Unique Cause

- **A condition is shown to independently affect a decision's outcome by varying just that condition while holding fixed all other possible conditions**



If the value of only one input toggles and the outcome toggles in response, then the cause is assumed to be the toggled input

- you don't need to see the internal logic of the expression to show independent effect

(Assume that the tests come from the requirements)

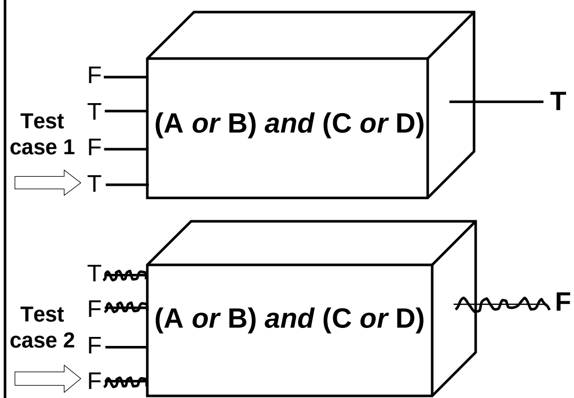
FAA National Software Conference - June 6, 2001 21

Masking

- **A condition is shown to independently affect a decision's outcome by using basic logic principles to assure that no other condition influences the outcome**
  - even though more than one condition may change value
- **In logical expressions, some inputs may hide or mask the effect of other inputs; for example,**
  - *false* ***and*** **X** is always *false*
  - *true* ***or*** **X** is always *true*
- **"Masking" principles are the converse**
  - *true* ***and*** **X** is **X**
  - *false* ***or*** **X** is **X**

FAA National Software Conference - June 6, 2001 22

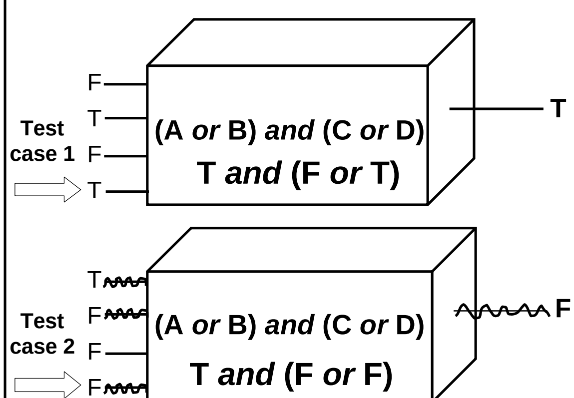
Masking



Analysis of the internal logic is needed to make sure that the condition of interest is the *only* toggled condition causing the decision's outcome to toggle.

FAA National Software Conference - June 6, 2001 23

Masking (cont.)



Analysis of the internal logic is needed to make sure that the condition of interest is the *only* toggled condition causing the decision's outcome to toggle.

D is the only input that affects the outcome

FAA National Software Conference - June 6, 2001 24

Acceptability of Masking MC/DC

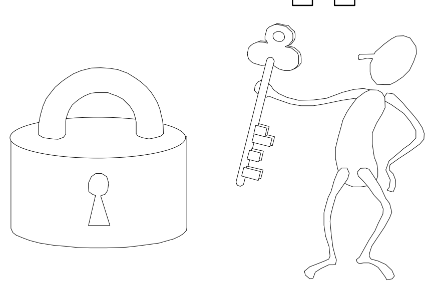
At the February 2001 meeting of the Certification Authorities Software Team (CAST), attendees concurred that masking MC/DC should be an acceptable means of meeting the MC/DC objective in DO-178B.

A paper titled "Rationale for Accepting Masking MC/DC in Certification Projects" has been submitted for CAST approval. The draft is included in the MC/DC tutorial.

FAA National Software Conference - June 6, 2001 25

Module 2

An MC/DC Approach



FAA National Software Conference - June 6, 2001 26

Module 2 Overview

- Defining Building Blocks for MC/DC
  - how to test basic logical constructs
- Using the Building Blocks for Decisions
- Building Block Approach to Evaluating MC/DC
- Examining Source Code
  - one line at a time
  - multiple lines

Minimum Tests

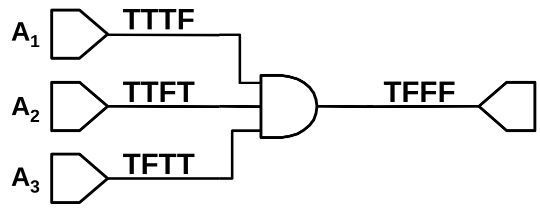
- MC/DC is intended to assure that each condition within a decision has been shown to have the proper effect
- Showing independent effect of a condition requires specific minimum tests for each logical operator
- Minimum tests provide the building blocks for assessing MC/DC

logical operator = logical gate

Testing an *n*-input *and* Gate

A_1 *and* A_2 *and* A_3 *and* … A_n

- **Minimum testing to provide MC/DC requires**
  - **all inputs *true*, output *true***
  - **each input individually *false*, output *false***
- **Example: 3-input *and* gate: TTT, TTF, TFT, FTT**

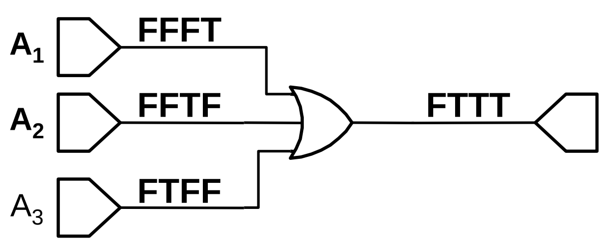


A_1 *and* A_2 *and* A_3

Testing an *n*-input *or* Gate

A_1 *or* A_2 *or* … A_n

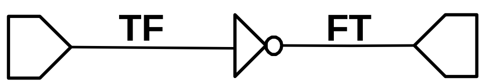
- **Minimum testing to provide MC/DC requires**
  - **all inputs *false*, output *false***
  - **each input individually *true*, output *true***
- **Example: 3-input *or* gate: FFF, FFT, FTF, TFF**



A_1 *or* A_2 *or* A_3

Testing a *not* Gate

- **Minimum testing to provide MC/DC requires**
  - **input *true*, output *false***
  - **input *false*, output *true***
- **Example:**



***not* A**

FAA National Software Conference - June 6, 2001 31

Testing an *xor* Gate

It's not like the other gates ...

- **More than one test set will satisfy the MC/DC criteria for an *xor* gate**
- **Minimum testing to provide MC/DC requires**
  - **any of the following for a 2-input *xor***
    - **TT, TF, FT**
    - **TF, FT, FF**
    - **FT, FF, TT**
    - **FF, TT, TF**

FAA National Software Conference - June 6, 2001 32

Question

Situation:

The software requirements call for evaluating the expression A *xor* B

The requirement is incorrectly implemented in source code as A *or* B

What test case is needed to catch the coding error?

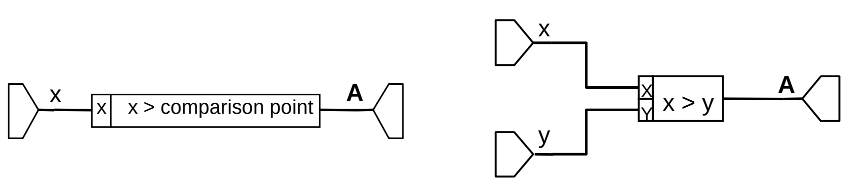
What do the Minimum Tests Provide?

- **For decisions with a common logical operator (e.g., A *and* B *and* C *and* …), the minimum tests guarantee that**
  - **the decision has taken all possible outcomes at least once**
  - **every condition has taken all possible outcomes at least once**
  - **every condition independently affects the decision outcome**

Now for some complex constructs

Testing a Comparator

- **A comparator evaluates 2 numerical inputs using a relational operator and returns a Boolean**



Leanna Rierson, Kelly Hayhurst, Dan Veerhusen

Comparator (cont.)

- **Minimum testing to provide MC/DC requires**
  - input x set at a value above the comparison point (or y)
  - input x set at a value below the comparison point (or y)

Example: (x < 5000)

Are these requirements sufficient to provide assurance that the logic is correct?

FAA National Software Conference - June 6, 2001 37

Minimum 'Good' Testing for a Comparator

- **Input x set at a value *slightly* above the comparison point**
- **Input x set at a value *slightly* below the comparison point**
- **Input x set at a value equal to the comparison point**

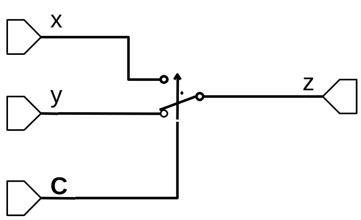
slightly depends on engineering judgement based on the numerical resolution of the target computer, the test equipment, and resolution of the output device

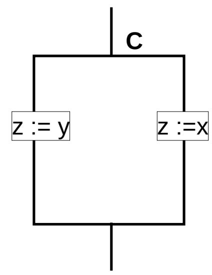
FAA National Software Conference - June 6, 2001 38

Testing an *if-then-else*

- **A Boolean expression controls the execution flow of an *if-then-else* statement**

Example: if C then z := x else z := y;





Minimum Testing for an *if-then-else*

(1) Inputs that force the execution of the *then* path

(2) Inputs that force the execution of the *else* path

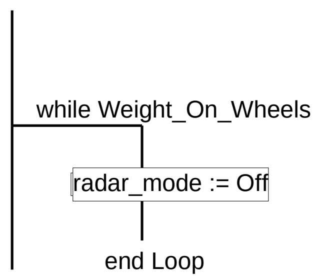
- **Note that the decision must evaluate to *false* with confirmation that the *then* path did not execute, even if there is no *else* path.**

(3) Inputs to exercise any logical gates in the decision using the minimum tests

Testing *while* loops

Example:

```
while Weight_On_Wheels loop
      radar_mode := Off;
end loop;
```



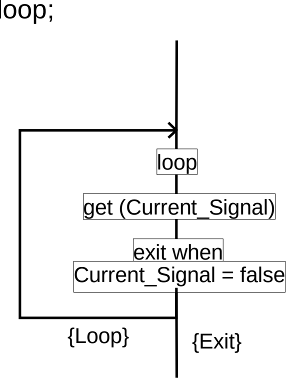
- **Minimum testing requires the following:**
  (1) inputs to force the execution of the statements in the loop
  (2) inputs to force the exit of the loop
  (3) inputs to exercise any logical gates in the decision using the minimum tests

FAA National Software Conference - June 6, 2001 41

Testing *exit when* loops

Example:

```
loop
   get (Current_Signal);
   exit when Current_Signal = false;
end loop;
```



- **Minimum testing requires the following:**
  (1) inputs to force the repeated execution of the statements in the loop when the decision for the *exit when* evaluates to *false*
  (2) inputs to force the immediate exit of the loop when the decision for the *exit when* evaluates to *true*
  (3) inputs to exercise any logical gates in the decision using the minimum tests

FAA National Software Conference - June 6, 2001 42

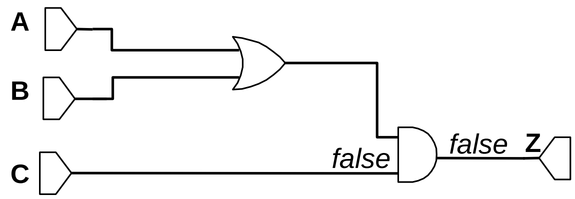
Applying the building blocks to determine whether a set of requirements-based test cases provide MC/DC of the source code

Evaluating MC/DC

- **For decisions with a common logical operator (A *or* B *or* C) or (A *and* B *and* C)**
  - **evaluating MC/DC requires checking the requirements-based tests to make sure they contain the minimum tests for that operator**
- **For decisions with mixed logical operators (A *or* B) *and* (C *or* D)**
  - **evaluating MC/DC is a bit more complicated**
  - **complications arise because one input to a logical operator may mask the effects of other inputs to that operator**

A Closer Look at Masking

Z:= (A or B) and C;



- ***false and*** X **is always** ***false***
- ***true or*** X **is always** ***true***

If you can't "see" the output of a gate for a particular test case, then that test case does not count towards coverage of that gate

Controllability & Observability

- **Basic concepts of testing logic circuits:**
  - **<u>controllability</u>: ability to control the inputs to a logical operator**
  - **<u>observability</u>: ability to observe the outputs of a logical operator at some end point**

5-Step Approach to Evaluating MC/DC

(1) Create a schematic representation of the source code

(2) Map the inputs of the requirements-based test cases to the schematic representation

(3) Eliminate masked test cases
   - those cases where the results for a specific gate are hidden from the observed outcome

(4) Determine MC/DC based on the building blocks for each logical operator

(5) Examine the outputs of the tests to confirm correct operation of the software

FAA National Software Conference - June 6, 2001 47

First, we will look at assessing MC/DC for a single line of source code.

Second, we will look at assessing MC/DC for multiple lines of source code.

FAA National Software Conference - June 6, 2001 48

Single Source Line Example

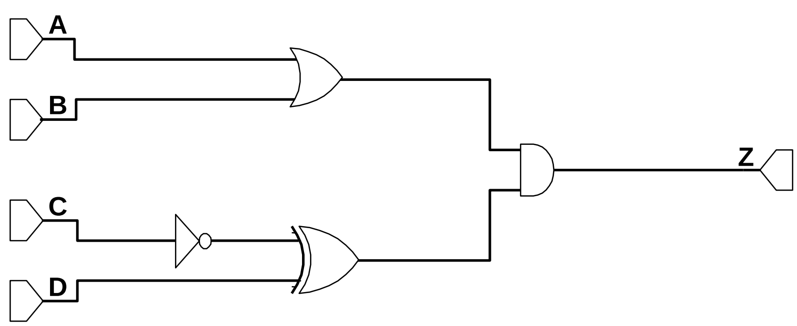
Z:= (A or B) and (not C xor D);

Requirements-based Test Cases

Test Case Number	1	2	3	4	5
A	T	F	F	T	T
B	F	F	T	T	T
C	T	T	T	F	F
D	T	T	T	T	F
Z	T	F	T	F	T

FAA National Software Conference - June 6, 2001 49

Step 1: Source Code Representation



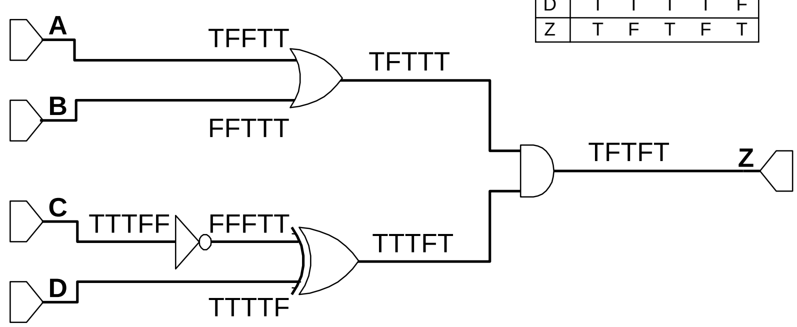
Z:= (A or B) and (not C xor D);

FAA National Software Conference - June 6, 2001 50

Step 2: Map Test Cases

Test Cases

	1	2	3	4	5
A	T	F	F	T	T
B	F	F	T	T	T
C	T	T	T	F	F
D	T	T	T	T	F
Z	T	F	T	F	T

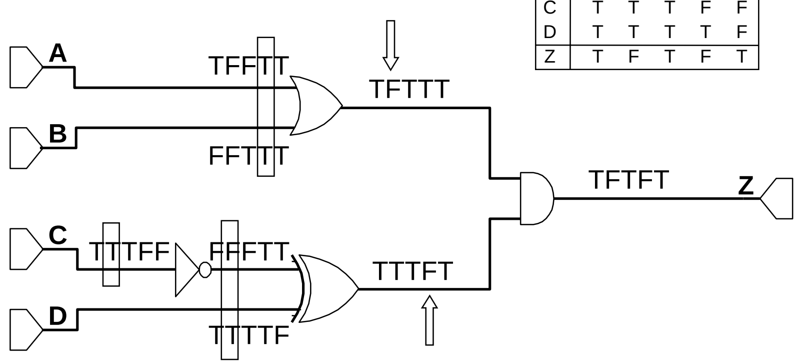


Z:= (A or B) and (not C xor D);

Step 3: Eliminate Masked Tests

Test Cases

	1	2	3	4	5
A	T	F	F	T	T
B	F	F	T	T	T
C	T	T	T	F	F
D	T	T	T	T	F
Z	T	F	T	F	T

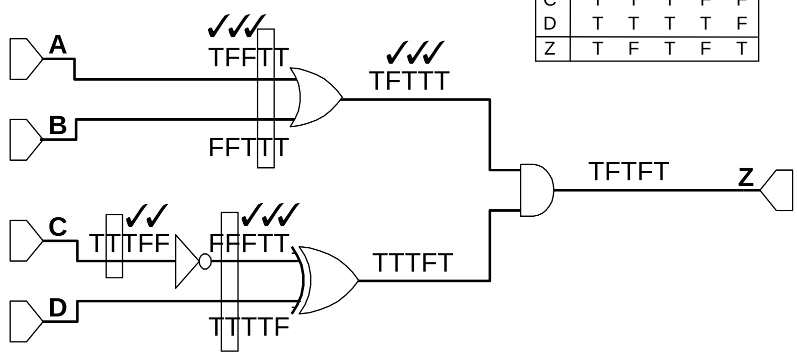


Z:= (A or B) and (not C xor D);

Step 4: Check for Minimum Tests

Test Cases

	1	2	3	4	5
A	T	F	F	T	T
B	F	F	T	T	T
C	T	T	T	F	F
D	T	T	T	T	F
Z	T	F	T	F	T



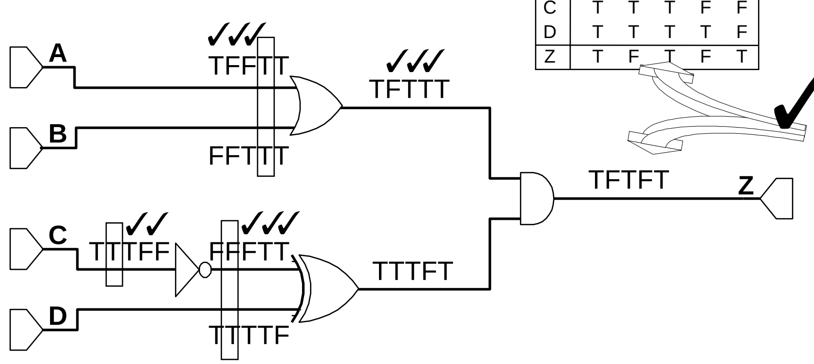
Z:= (A or B) and (not C xor D);

FAA National Software Conference - June 6, 2001 53

Step 5: Confirm Final Results

Test Cases

	1	2	3	4	5
A	T	F	F	T	T
B	F	F	T	T	T
C	T	T	T	F	F
D	T	T	T	T	F
Z	T	F	T	F	T



Z:= (A or B) and (not C xor D);

FAA National Software Conference - June 6, 2001 54

Multiple Source Lines Example

Requirement: Perform a voting operation on three input Booleans where the output is to be *true* whenever at least two of the inputs are *true*.

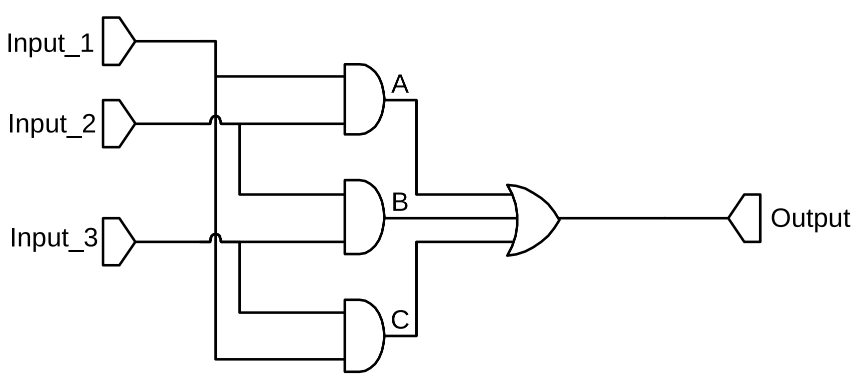
Source Code:

```
A := Input_1 and Input_2;
B := Input_2 and Input_3;
C := Input_3 and Input_1;
Output := A or B or C;
```

Requirements-based Test Cases

Test Case Number	1	2	3	4
Input_1	T	T	F	F
Input_2	T	F	T	F
Input _3	F	T	T	F
Output	T	T	T	F

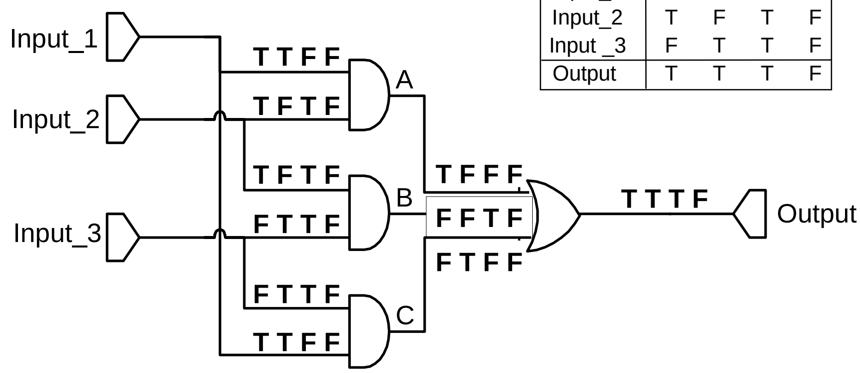
Step 1: Source Code Representation



Step 2: Map Test Cases

Test Cases

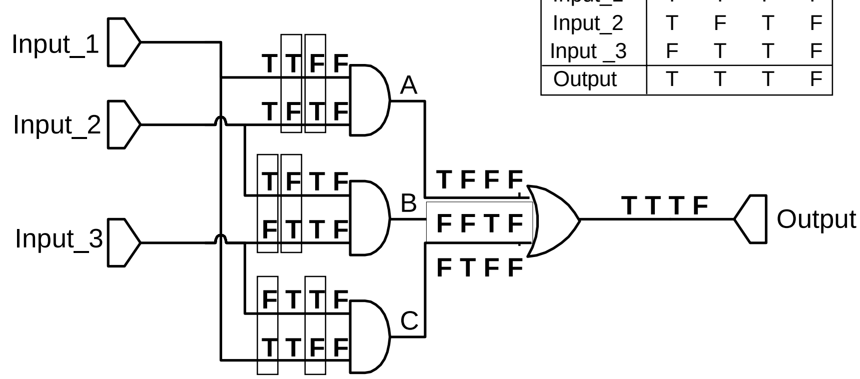
	1	2	3	4
Input_1	T	T	F	F
Input_2	T	F	T	F
Input _3	F	T	T	F
Output	T	T	T	F



Step 3: Eliminate Masked Tests

Test Cases

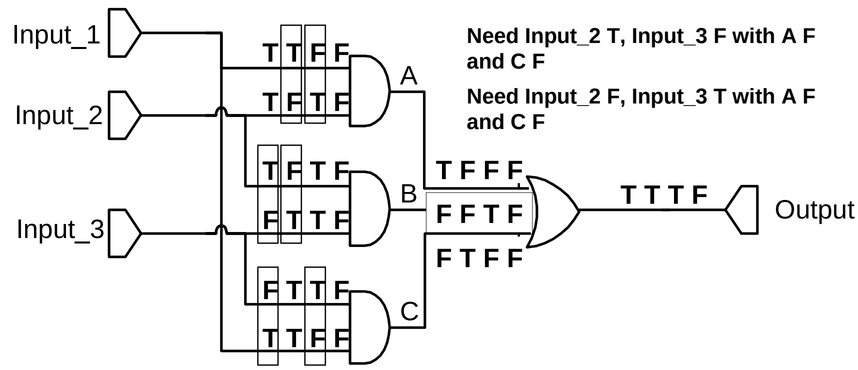
	1	2	3	4
Input_1	T	T	F	F
Input_2	T	F	T	F
Input _3	F	T	T	F
Output	T	T	T	F



Leanna Rierson, Kelly Hayhurst, Dan Veerhusen

Step 4: Check for Minimum Tests

Need Input_1 T, Input_2 F with B F and C F
Need Input_1 F, Input_2 T with B F and C F



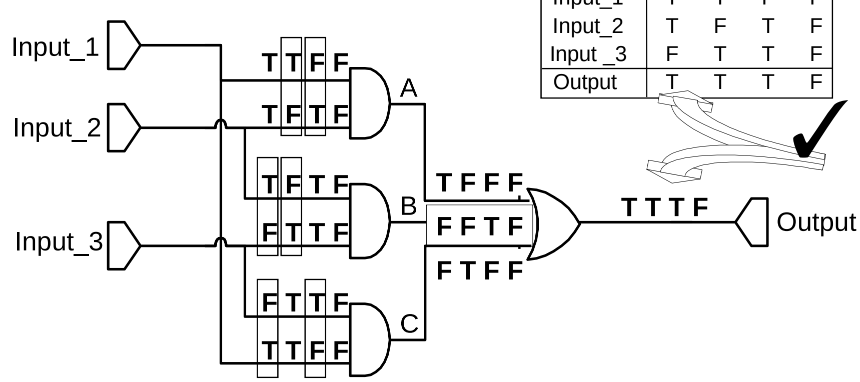
Need Input_1 T, Input_3 F with A F and B F
Need Input_1 F, Input_3 T with A F and B F

FAA National Software Conference - June 6, 2001 59

Step 5: Confirm Results

Test Cases

	1	2	3	4
Input_1	T	T	F	F
Input_2	T	F	T	F
Input _3	F	T	T	F
Output	T	T	T	F



FAA National Software Conference - June 6, 2001 60

Analysis Resolution

- **Coverage analysis can reveal that the code structure was not exercised sufficiently by the requirements-based test cases**
  - **inadequate requirements-based tests or procedures**
  - **inadequate software requirements**
  - **dead or deactivated code**
- **Section 6.4.4.3 of DO-178B provides guidance for each of these**

FAA National Software Conference - June 6, 2001 61

Analysis Resolution (cont.)

- **Coverage analysis may also identify errors in the source code**
  - **there may be an error even if the actual results match the expected results**

<u>Example:</u>

Requirement: **A and (B xor C)**
Requirements-based Test Cases

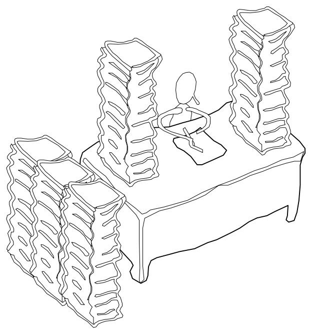
Test Case Number	1	2	3	4
A	F	T	T	T
B	F	F	T	T
C	T	F	F	T
Output	F	F	T	F

Source Code: **B and (B xor C)**

- **Expected results will match the actual results**
- **MC/DC analysis will show that the *xor* gate is not adequately tested**
  - **further analysis will show the mismatch between requirements and code**

FAA National Software Conference - June 6, 2001 62

Module 3

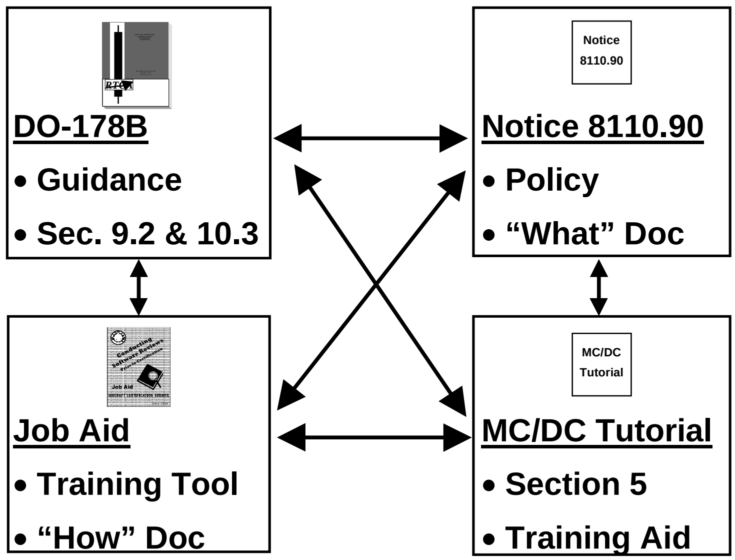


MC/DC Compliance Assessment

Module 3 Overview

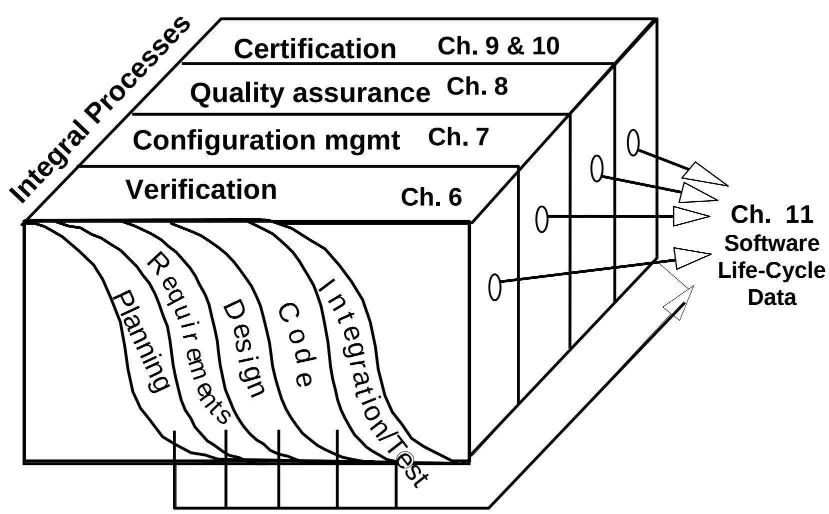
- **Assessment policy/guidance**
- **How MC/DC assessment fits into the review process**
- **Steps for assessing:**
  - **Verification plans**
  - **Tool qualification data**
  - **Test cases and procedures**
  - **Effectiveness of test program**

Assessment Policy/Guidance



FAA National Software Conference - June 6, 2001 65

DO-178B Layout



FAA National Software Conference - June 6, 2001 66

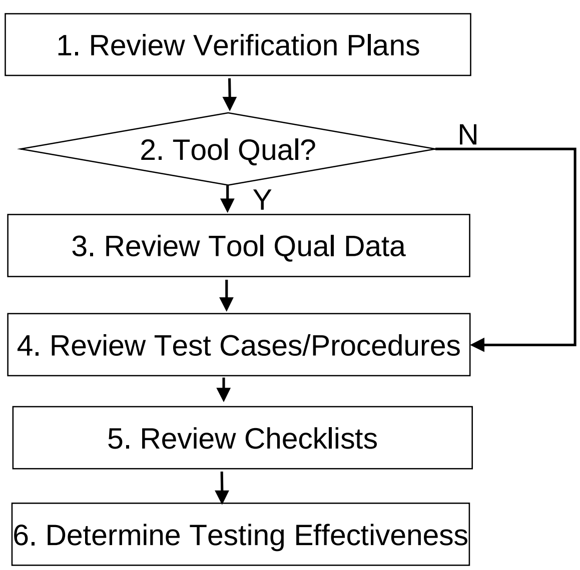
How MC/DC Fits Into the Review Process

- **Occurs during Stages of Involvement #1 & #3**
- **Occurs while reviewing for compliance to DO-178B Tables A-6 and A-7**

- **Purpose of Structural Coverage:**
  - **Show thoroughness of requirements-based testing**
  - **Assure absence of unintended functionality and dead code**
  - **Assure adequacy of code structure verification**

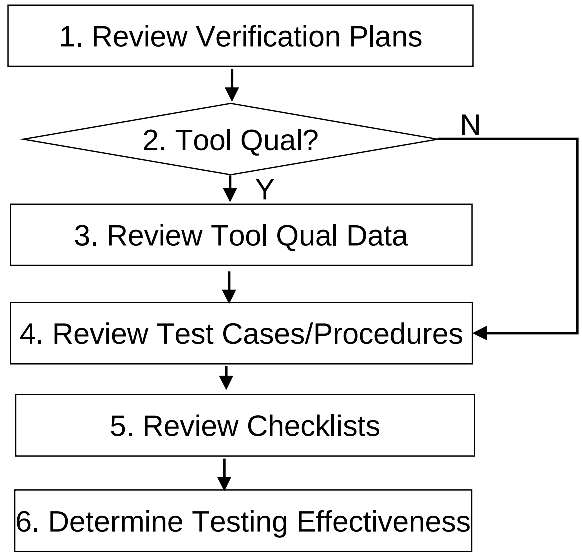
FAA National Software Conference - June 6, 2001 67

Assessment Process (6 Steps)



FAA National Software Conference - June 6, 2001 68

Assessment Process (6 Steps)



Step 1 – Review Verification Plans

- **Software Verification Plan (SVP) – Contains bulk of verification planning information.**
- **Purpose of SVP – Provides verification team with project-specific information needed for their job. And, assures that, if followed, applicable DO-178B objectives will be met.**
- **PSAC, CM Plan, and QA Plan might also provide some verification planning information.**
- **Plans are reviewed in Stage of Involvement (SOI) #1.**

Contents of SVP

- Developer's organization
- Definitions of independence for the project
- Verification methods and environment
- Transition criteria for verification
- Re-verification guidelines
- Odds and ends
  - Partitioning considerations
  - Previously developed software
  - Verification tools

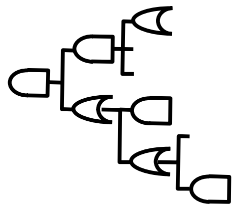
FAA National Software Conference - June 6, 2001 71

Step 1 Questions

- Are plans clear and detailed?
- Are roles of verification team members clear?
- Is requirements test level specified?
- Is the change process and needed regression analysis/testing addressed?
- Is tool reuse addressed?

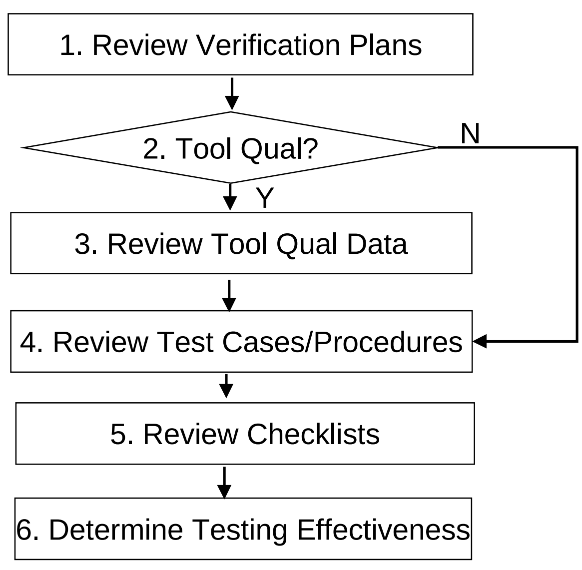
FAA National Software Conference - June 6, 2001 72

Step 1 Questions (cont.)

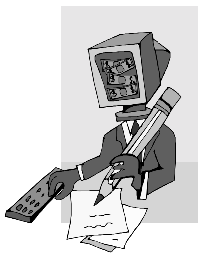


- **Is the MC/DC approach addressed in the plans?**
  - **Tools used for MC/DC**
  - **Relationship to requirements-based tests**
  - **Process for adding additional tests, if coverage isn't achieved**
  - **Transition criteria**
  - **Regression analysis/testing**

Assessment Process (6 Steps)



Step 2 – Determine if Tool Qualification is Needed

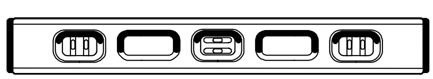


- Notice 8110.91 (previously 8110.83) documents much of the information for determining:
  - If tool qualification is needed
  - What data is needed to support tool qualification
  - Acceptance criteria for the Tool Operational Requirements document

FAA National Software Conference - June 6, 2001 75

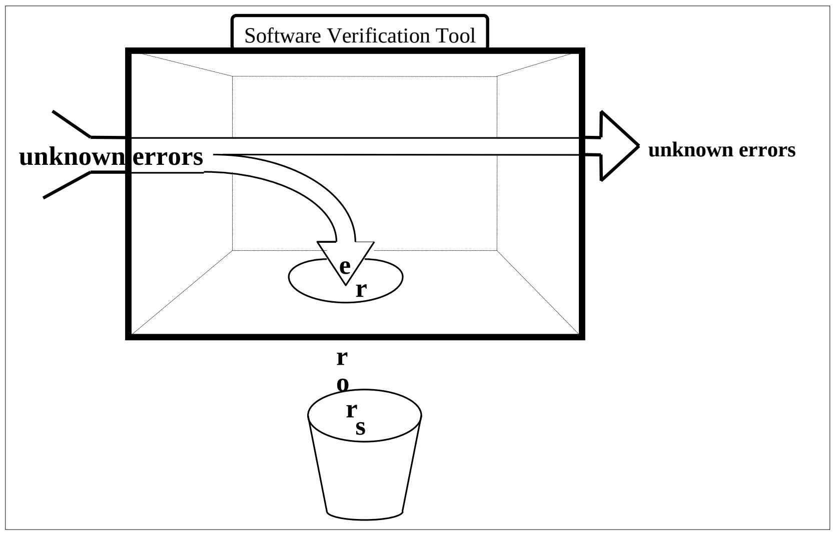
Verification Tools

- **Tools that cannot introduce errors, but may** fail to detect **them.**
  - **For example, a static analyzer, which automates a software verification process activity, should be qualified** if the function **that it performs is** not verified **by another activity. Type checkers, analysis tools and test tools are other examples.**
  - **MC/DC tools are typically considered verification tools.**



FAA National Software Conference - June 6, 2001 76

Verification Tools (cont.)

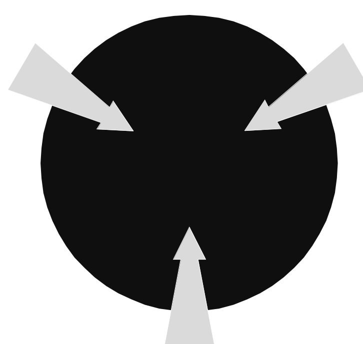


FAA National Software Conference - June 6, 2001 77

Three Questions To Determine If Verification Tools Need to Be Qualified

1. Can tool allow an existing error to remain undetected?

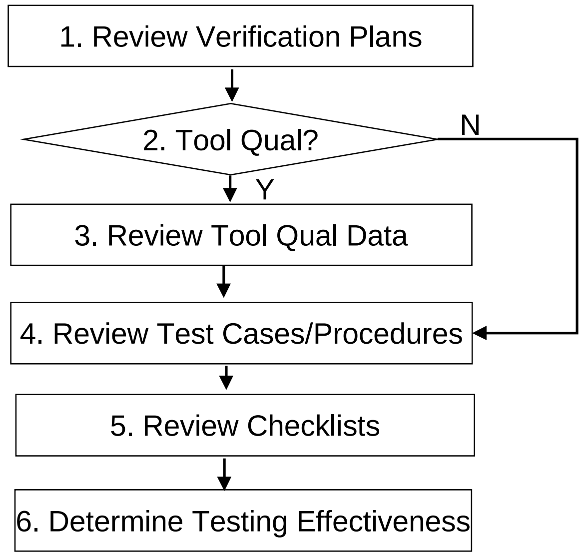
2. Will tool's output not be verified per section 6 of DO-178B?



3. Are processes required by DO-178B objectives eliminated, reduced, or automated?

FAA National Software Conference - June 6, 2001 78

Assessment Process (6 Steps)



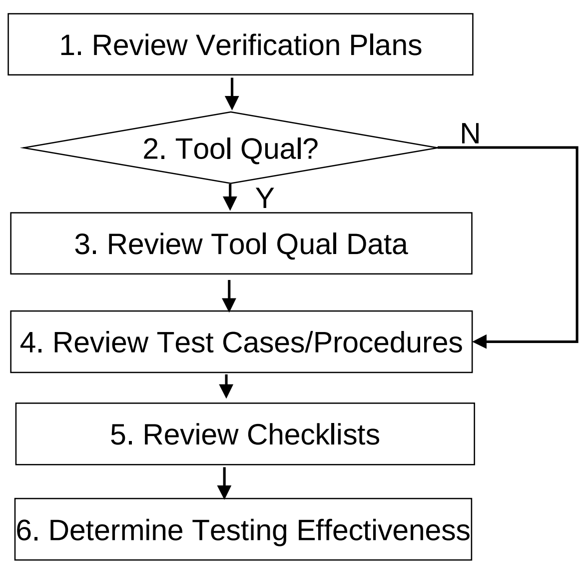
Step 3 – Review MC/DC Tool Qualification Data (if required)

- **Tool Qualification Data:**
  - **PSAC or Tool Qualification Plan**
  - **Tool Operational Requirements**
    - **Functionality**
    - **Operational Environment**
    - **Installation or Operational Information**
  - **Tool Verification Data**
    - **Normal Operating Conditions**
    - **Only test used portions of the tool**
  - **Software Accomplishment Summary or Tool Accomplishment Summary**

Step 3 Questions

- **Is tool qualification info included in plans?**
- **Does Tool Operational Requirements document all tool functions?**
- **Is instrumentation of code addressed (if needed)?**
- **Is the tool qual process sufficient to detect errors in the tool?**
- **Are tool procedures documented?**
- **Is the change process for the tool documented?**
- **Is the tool configuration documented and used?**
- **Are tool limitations clearly documented?**

Assessment Process (6 Steps)



Step 4 – Review Test Cases/Procedures

- Purpose of structural coverage:
  - Show thoroughness of requirements-based testing
  - Assure absence of unintended functionality and dead code
  - Assure adequacy of code structure verification
- Ideally, much of the structural coverage is obtained through requirements-based testing.
- Review of the overall verification program (particularly the test portion) is needed to understand the context of the MC/DC analysis.

FAA National Software Conference - June 6, 2001 83

Step 4 Questions

- Have verification plans been followed?
- Is the rationale for each test case/procedure clear?
- Are test cases/procedures traceable to the requirements?
- Have test cases/procedures been verified (see Step 5 for checklist information)?
- Do the test cases/procedures specify required input & expected output data?
- Are test cases/procedures sufficient to meet MC/DC?

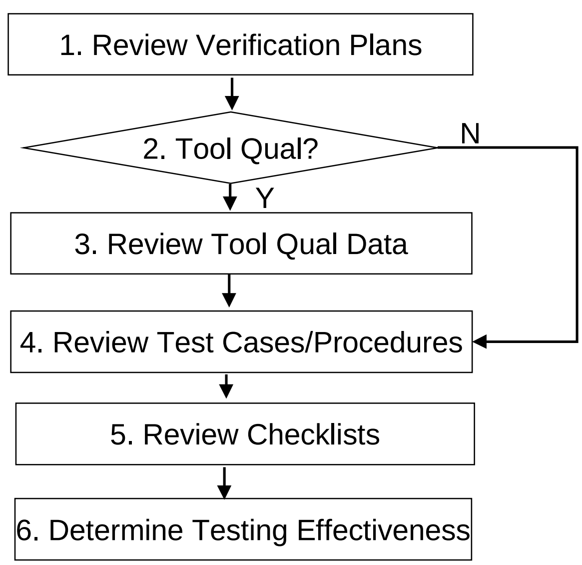
FAA National Software Conference - June 6, 2001 84

Step 4 Questions (cont.)

- **Are there sufficient robustness test cases/procedures?**
- **Are test cases/procedures correct?**

FAA National Software Conference - June 6, 2001 85

Assessment Process (6 Steps)



FAA National Software Conference - June 6, 2001 86

Use of Checklists

Table A-7
Verification Of Verification Process Results

	Objective		Applicability by SW Level				Output		Control Category by SW level			
	Description	Ref	A	B	C	D	Description	Ref	A	B	C	D
1	Test procedures are correct.	6.3.6b	○	○	○		Software Verification Results	11.14	②	②	②	
2	Test results are correct and discrepancies explained.	6.3.6c	○	○	○		Software Verification Results	11.14	②	②	②	
3	Test coverage of high-level requirements is achieved.	6.4.4.1	○	○	○	○	Software Verification Results	11.14	②	②	②	②
4												

Many of the objectives of Table A-7 are met by review/analysis using checklists.

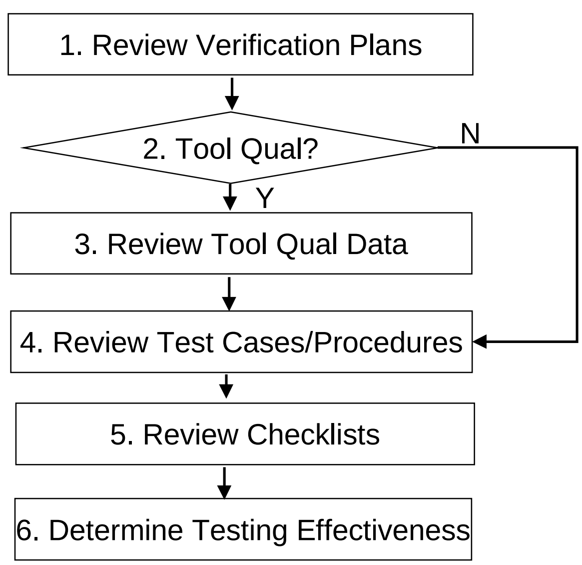
Step 5 Questions

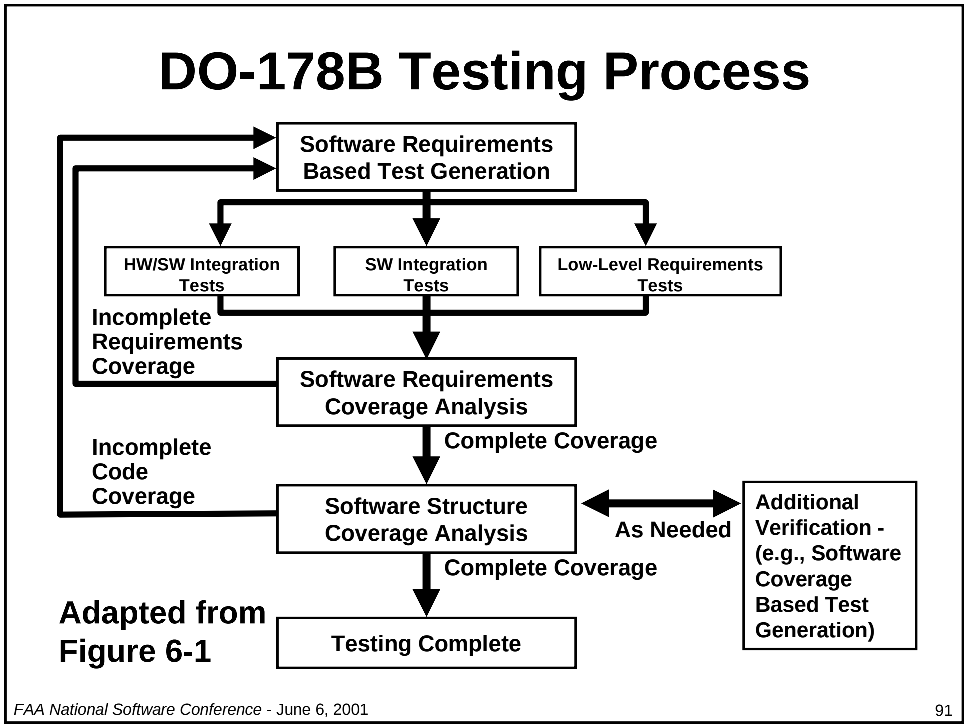
- **Are checklists sufficient to assure that MC/DC objective is met?**
- **Have checklists been reviewed and followed?**
- **Do checklists have the following specifics:**
  - **Who performed the review**
  - **What data was reviewed**
  - **When the data was reviewed**
  - **Review Results**
  - **Required Corrective Action**

Step 5 Questions (cont.)

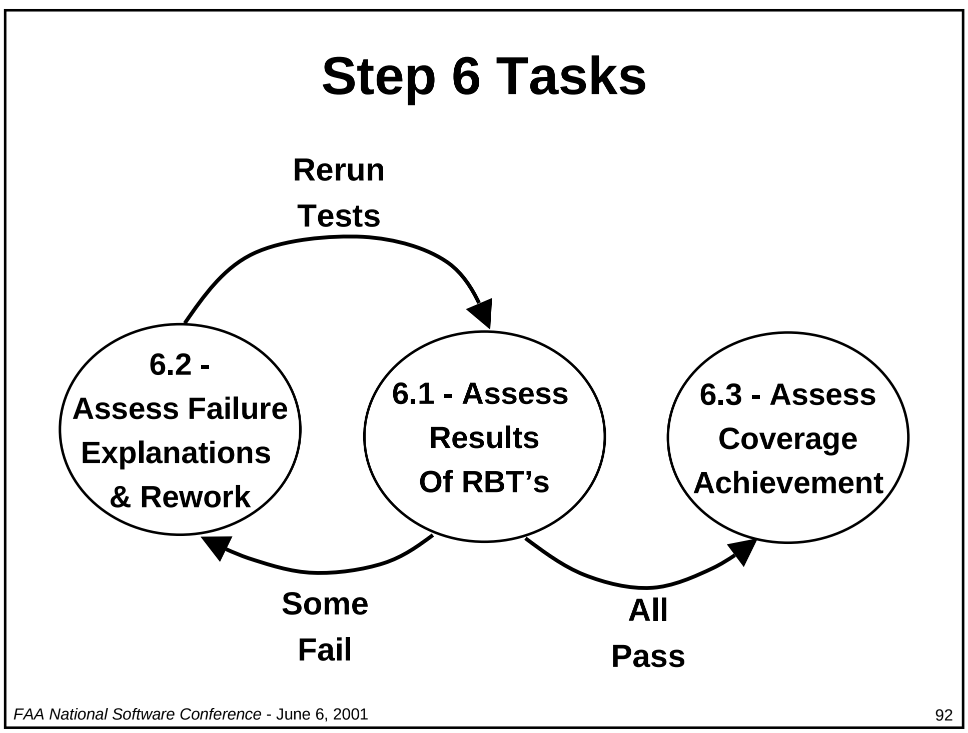
- **Do checklists ensure visual verification of test results?**
- **Will checklists address limitations of tools?**
- **Will checklists review when plans or standards have not been followed?**
- **Will checklists reveal when 100% structural coverage is not expected?**

Assessment Process (6 Steps)





DO-178B Testing Process
Software Requirements Based Test Generation
HW/SW Integration Tests
SW Integration Tests
Low-Level Requirements Tests
Incomplete Requirements Coverage
Software Requirements Coverage Analysis
Complete Coverage
Incomplete Code Coverage
Software Structure Coverage Analysis
As Needed
Additional Verification - (e.g., Software Coverage Based Test Generation)
Complete Coverage
Adapted from Figure 6-1
Testing Complete
FAA National Software Conference - June 6, 2001
91



Step 6 Tasks
Rerun Tests
6.2 - Assess Failure Explanations & Rework
6.1 - Assess Results Of RBT's
6.3 - Assess Coverage Achievement
Some Fail
All Pass
FAA National Software Conference - June 6, 2001
92

Task 6.1 – Assess RBT Results

- Is each test result clearly linked to a test case?
- Are failed test cases obvious from the results?
- Has configuration control been implemented?
- Do test results adhere to plans and procedures?

FAA National Software Conference - June 6, 2001 93

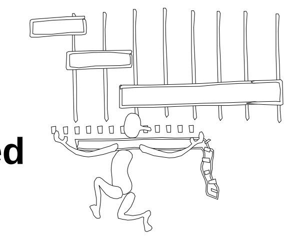
Task 6.2 – Assess Failure Explanations and Rework

- Is explanation of failed tests cases technically sound and accurate?
- Is there a reference to relevant problem reports?
- Is rework of test cases or code adequately explained?
- Have test cases been re-run and test results recorded per regression plans?

FAA National Software Conference - June 6, 2001 94

Task 6.3 – Assess Coverage Achievement

- Has MC/DC criteria been correctly applied?
- Is MC/DC achieved through RBTs?
- If not, has the rationale been documented?
- Are statement and decision coverage also achieved?
- Are drops in coverage explained sufficiently?
- Has dead code been identified and analyzed/removed?



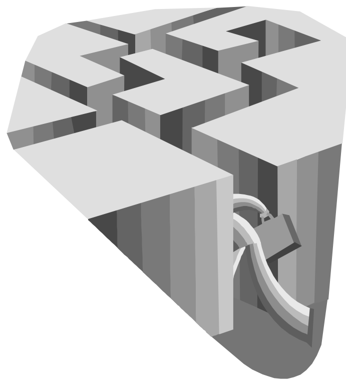
FAA National Software Conference - June 6, 2001 95

Module 4

Common Pitfalls

FAA National Software Conference - June 6, 2001 96

Module 5 Overview



There are lots of wrong turns that can be taken.

FAA National Software Conference - June 6, 2001 97

Inadequate Planning

- **Not enough detail in the verification plans**
- **Failure to plan the change process**
- **Inadequate tool qualification planning**
- **Failure to follow the plans**
- **Tendency to not update plans as process changes occur**
- **Failure to consider importance of coding & design standards on verification effort (e.g., complexity, tight coupling)**

FAA National Software Conference - June 6, 2001 98

Misunderstanding the MC/DC Objective

- Trying to meet MC/DC separate from requirements-based testing
- Uncertainty about source vs. object code coverage
- Lack of understanding of structural coverage intent
- Trying to meet MC/DC too early
- Using MC/DC as a testing method

FAA National Software Conference - June 6, 2001 99

Misunderstanding the MC/DC Objective (cont.)

- Not recognizing tie between MC/DC and code standards and compiler settings
- Expecting too much of structural coverage tools
- Not knowing when MC/DC has been achieved

FAA National Software Conference - June 6, 2001 100

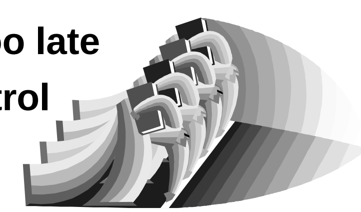
Ineffective Test Strategies

- Complex software change tracking
- Redundant activities
- Conflicting results from tools

FAA National Software Conference - June 6, 2001 101

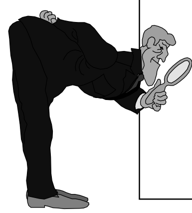
Poor Management of Verification Resources

- Inexperienced/unqualified verification engineers
- Inadequate training of verification team
- Inadequate documentation of test cases/procedures [makes future changes difficult]
- Critical feature tested too late
- Inadequate change control
- Over-reliance on tools



FAA National Software Conference - June 6, 2001 102

What's A Body To Do?

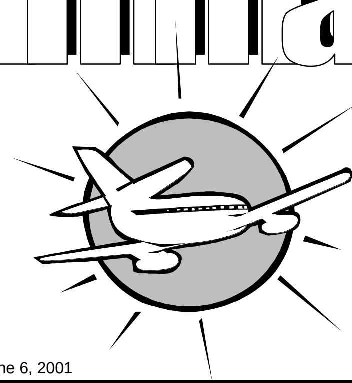


- Cert authorities & Designees:
  - Be alert to these potential pitfalls
  - Encourage early planning & cert liaison
  - Realize that poor planning and implementation can often lead to non-compliance

- Applicants:
  - Take steps to mitigate these problems
  - Make plans & processes work for you

FAA National Software Conference - June 6, 2001 103

MC/DC Summary



FAA National Software Conference - June 6, 2001 104

Major Topics

- **Module 1**
  - **Defined MC/DC**
- **Module 2**
  - **Provided an approach to MC/DC evaluation**
- **Module 3**
  - **Addressed assessment of MC/DC data**
- **Module 4**
  - **Discussed common problems with MC/DC**

FAA National Software Conference - June 6, 2001 105

Additional Information on MC/DC

- **The MC/DC tutorial**
- **FAA Web-site**
  - **http://av-info.faa.gov/software**
- **Research report**
- **DO-178B and DO-248[ ]**
- **Certification Authorities Software Team (CAST) paper**
- **Feel free to send questions**

FAA National Software Conference - June 6, 2001 106