SAFRAN ENGINEERING SERVICES PROGRAMMING UNDER DO178 - 332

Olivier DUPOUY

DO-178 OVERVIEW

SAFRAN

DO-178 Overview

- DO-178 - Software Considerations in Airborne Systems and Equipment Certification
- Standard of RTCA Incorporation (in Europe it is ED-12B and standard of EUROCAE)
- Represents the avionics industry consensus to ensure software safety
- Acceptable by FAA and EASA certification authorities
- "The FAA and the civil aviation community recognize RTCA'S DO-178B as an acceptable means of compliance to the FAA regulations for SW aspects of certification."

Software Levels in DO-178

Different failure conditions require different software conditions -> 5 levels

Failure Condition	Software Level
Catastrophic	Level A
Hazardous/Severe - Major	Level B
Major	Level C
Minor	Level D
No Effect	Level E

SAFRAN

Examples DO-178 Safety Levels

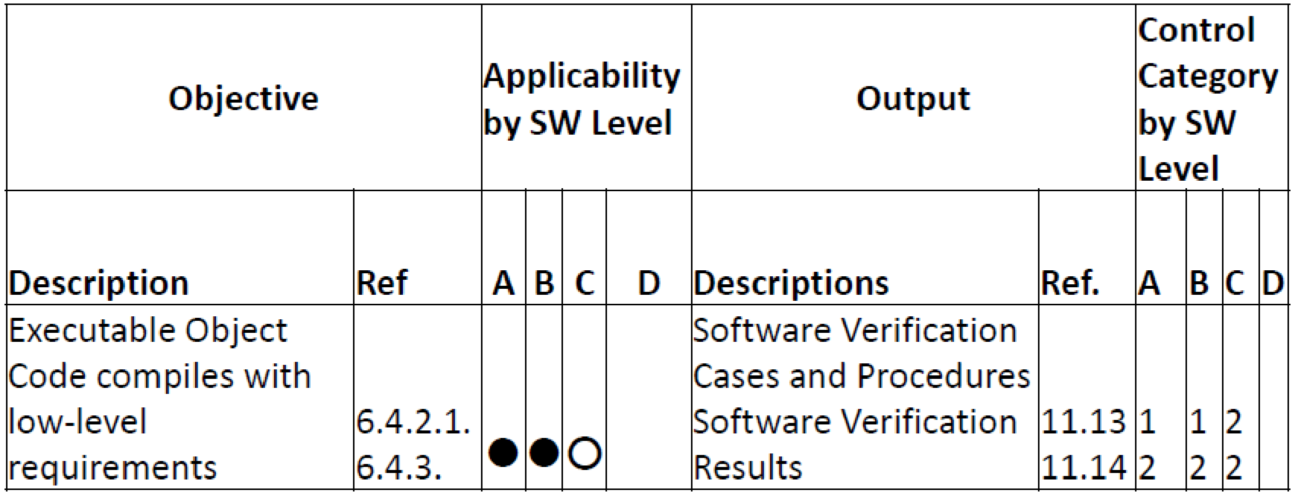
Safety-critical Levels C&D	Safety-critical Levels A&B
Anti-missile defense	Fly-by-wire controls
Data mining	Auto-pilot
Health monitoring	Air-traffic Separation Control
Mission planning and implementation	Glass Cockpit Information Display
Mission simulation and training	Radar
Network-centric operation	Jet Engine Control
Real-time data recording and analysis	IFF (friend or foe)
Self-healing communication networks	Missile guidance
Telemetry	Missile launch
Weapons targeting	Missile self-destruct

Objectives for Safety Levels

Different levels of safety requires different objectives to be fulfilled

Defined by some tables in ANNEX A

Example: Table A-6 Objective 3.



Objective		Applicability by SW Level				Output		Control Category by SW Level			
Description	Ref	A	B	C	D	Descriptions	Ref.	A	B	C	D
Executable Object Code compiles with low-level requirements	6.4.2.1. 6.4.3.	●	●	○		Software Verification Cases and Procedures Software Verification Results	11.13 11.14	1 2	1 2	2 2	

SW DEVELOPMENT

The plans

Five different plans:

- SW Development Plan
- SW Verification Plan
- SW Quality Assurance Plan
- SW Configuration Plan
- SW Aspects of Certification

Verification, management, quality assurance and certification are overlaid on the defined development process

Software Planning

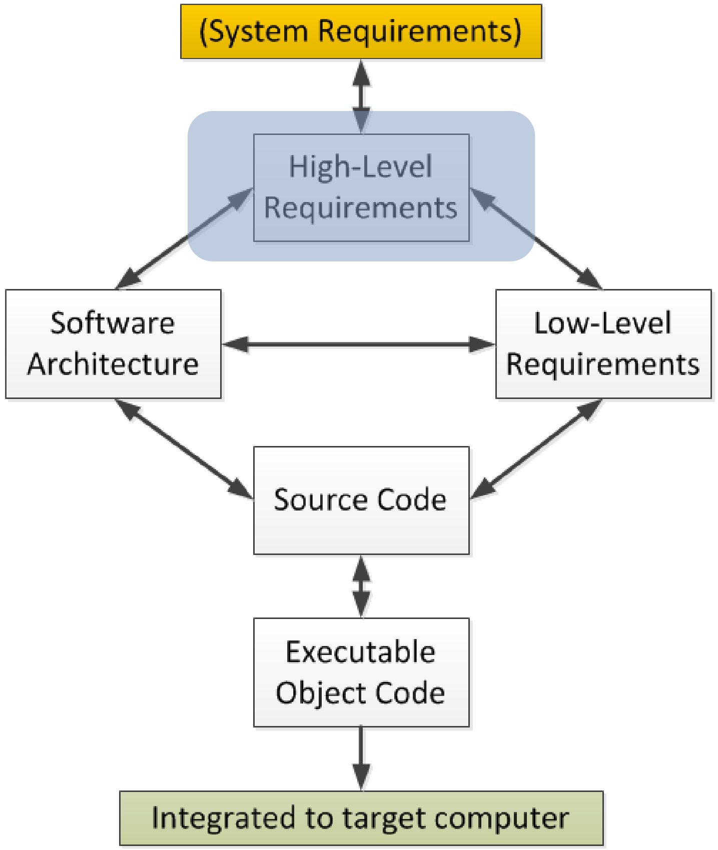
SW development standards

- ◆ SW requirements standard
  - > Language to be used (Plain English…)
- ◆ SW design standards
  - > Complexity limits, exclusion of recursion, dynamic memory allocation
- ◆ SW Code standards
  - > Syntax, semantics and constraints

SW Development

High-Level requirements

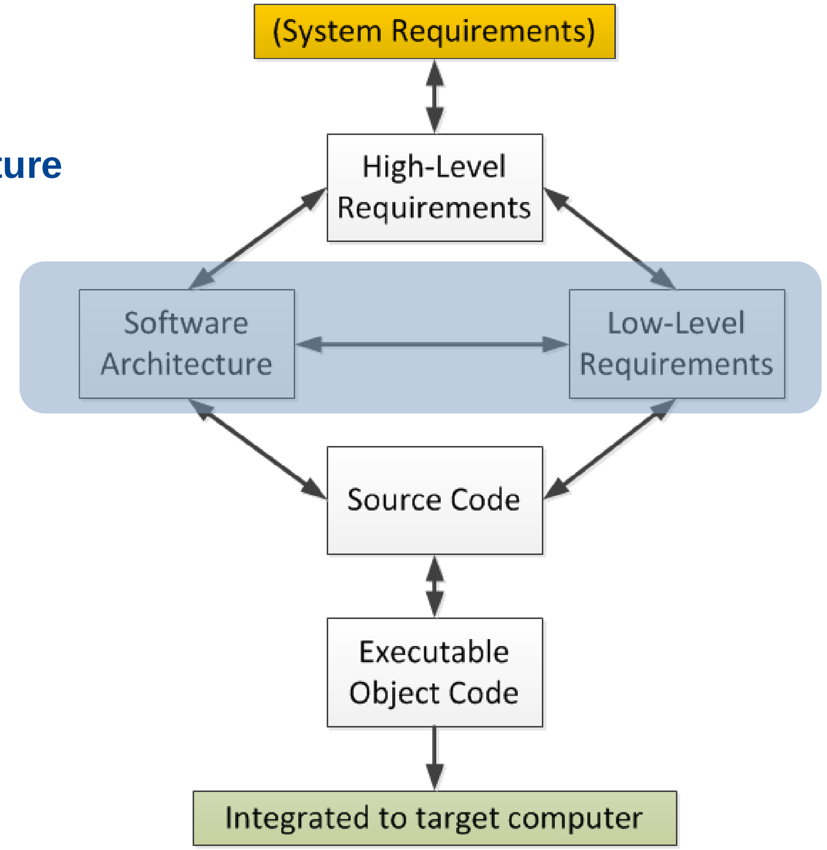
- Based on system analysis and safety assessment
- Black-box view of the software component
- System level considerations
- Functional requirements by mode of operation
- Performance criteria
- Timing requirements
- Memory size constraints
- HW and SW interfaces



SW Development

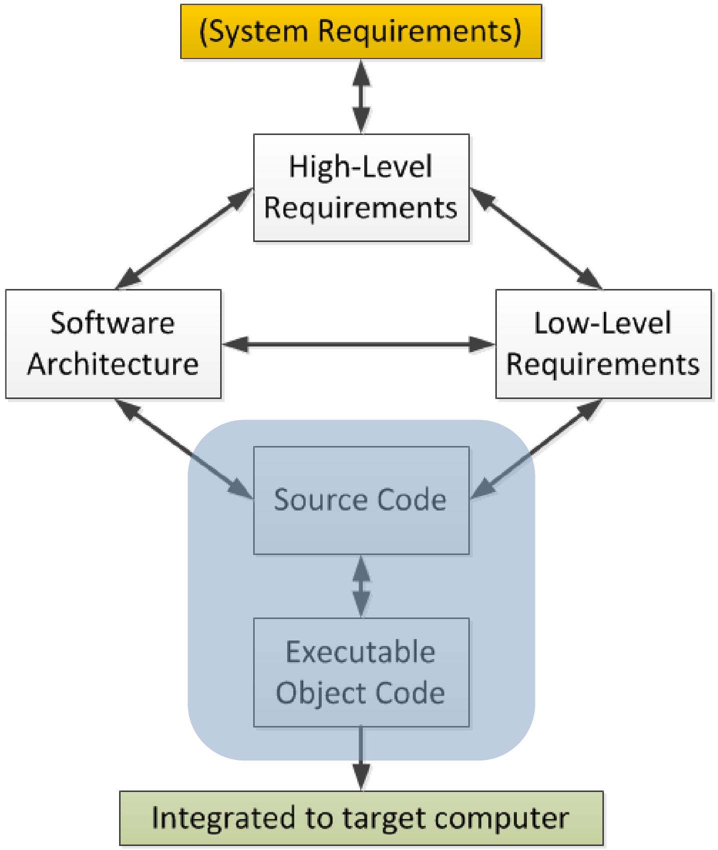
Low-Level requirements and Software Architecture

- SW requirements
- Derived from High-Level requirements
- Design constraints
  - Task allocation
  - Algorithms
  - Data Structures
- Input/output definitions
- Data and Control flows
- Resource management and scheduling
- Design Methods



SW Development

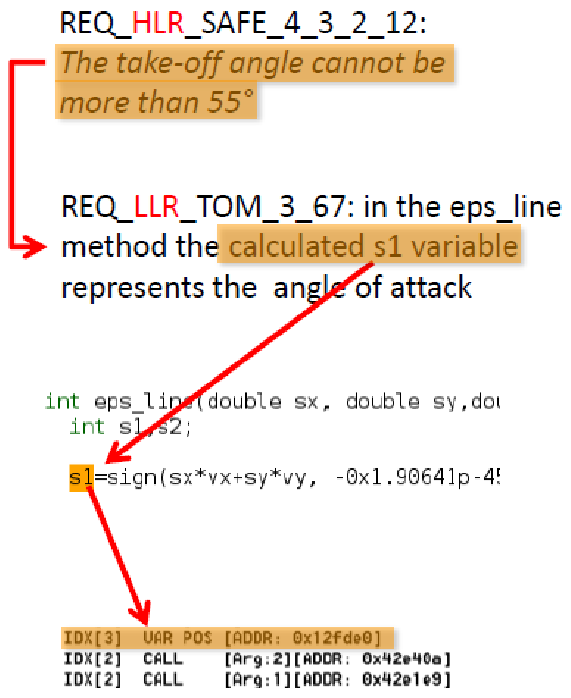
- Source Code
  - Usually collection of “high-level” language and assembly
  - Includes linker files, compile commands etc.
- Executable
  - Completely target computer specific
  - “machine readable”
- Final output is the integrated system on the target platform
  - Cf. DO254



TRACEABILITY & VERIFICATION

Traceability

- Through the complete product life-cycle (30+ years)
- From requirements to byte code (Level A)
- Essential for maintainability
- Back-annotation of errors
- Typical implementation:
  - Excel
  - *Doors*
  - *PTC Integrity*
- Code generators usually gives extensive support
- Hard in case of multiple development tools



REQ_HLR_SAFE_4_3_2_12:
The take-off angle cannot be more than 55°

REQ_LLR_TOM_3_67: in the eps_line method the calculated s1 variable represents the angle of attack

```
int eps_line(double sx, double sy,dou
  int s1,s2;

  s1=sign(sx*vx+sy*vy, -0x1.90641p-45
```

```
IDX[3]  VAR POS [ADDR: 0x12fde0]
IDX[2]  CALL    [Arg:2][ADDR: 0x42e40a]
IDX[2]  CALL    [Arg:1][ADDR: 0x42e1e9]
```

Verification

- Two purposes
  - Demonstrate intended function
  - Demonstrate (to the extent possible) the absence of unintended function
- Consists of
  - Reviews
  - Analysis
  - Testing
- The FAA or EASA representative needs to accept all part of the verification process. (e.g., test cases)
- Certification has legal issues : the developer is responsible!

Verification

Reviews:

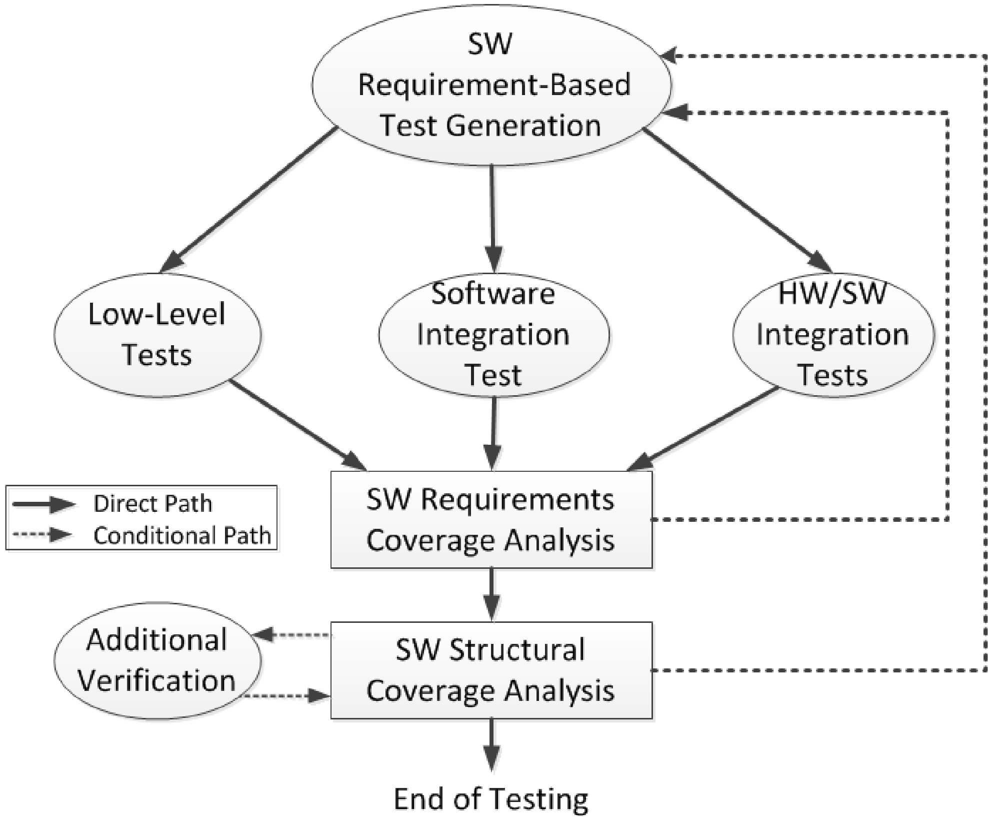
- Qualitative assessment of the process or product
- Typical implementation: checklist
- Applied on all SW Development process step (HLR, LLR, SW Arch., SW Coding, Test cases, etc.)

Analysis:

- Provide repeatable evidence of correctness
- Typical implementation: timing, stack analysis, data flow and call-tree

Verification – Testing

Testing:

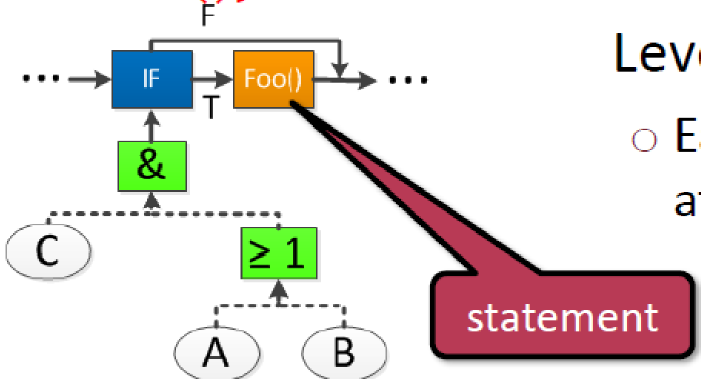

SW Requirement-Based Test Generation
Low-Level Tests
Software Integration Test
HW/SW Integration Tests
Direct Path
Conditional Path
SW Requirements Coverage Analysis
Additional Verification
SW Structural Coverage Analysis
End of Testing

Verification – Testing

- Structural Coverage
  - Determine what software structure were not exercised
- Levels:
  - Statement Coverage
  - Decision Coverage
  - Modified Decision / Condition Coverage (MC/DC)
    - Each decision tries every possible outcome
    - Each condition in a decision takes on every possible outcome
    - Each entry and exit point is invoked
    - Each condition in a decision is shown to independently affect the outcome of the decision
- Gaps
  - Complier induced code (e.g., array bound checks)
  - Deactivated code
  - Dead code
- Performed on source code,
  - except Level A
    - Correspondence must be shown
    - Complier optimization can introduce new code
- In addition, coverage of data and control coupling is required

Verification – Testing examples (1/3)

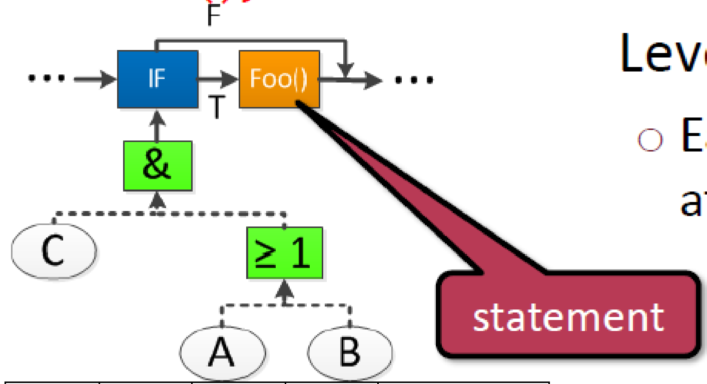
```
IF(C AND( A OR B))
   THEN Foo();
```



- Statement Coverage (SC) Level C
  - Each statement is executed at least once

Verification – Testing examples (1/3)

```
IF(C AND( A OR B))
   THEN Foo();
```

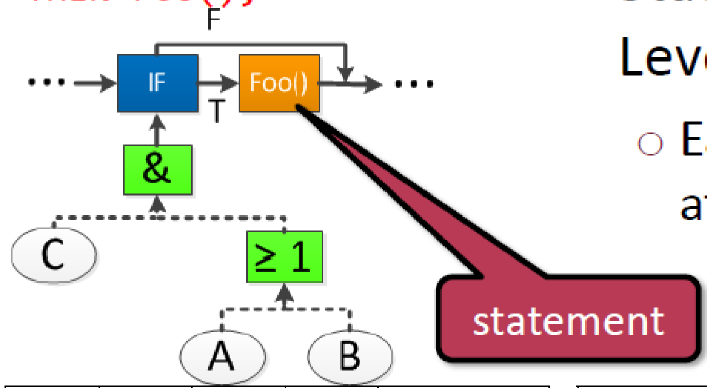


#	A	B	C	Foo Executed
1	0	0	0	NO
2	0	0	1	NO
3	0	1	0	NO
4	0	1	1	YES
5	1	0	0	NO
6	1	0	1	YES
7	1	1	0	NO
8	1	1	1	YES

- Statement Coverage (SC) Level C
  - Each statement is executed at least once

Verification – Testing examples (1/3)

```
IF(C AND( A OR B))
   THEN Foo();
```



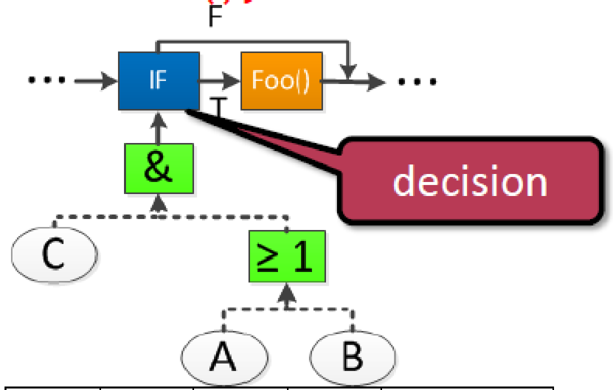
- Statement Coverage (SC) Level C
  - Each statement is executed at least once

#	A	B	C	Foo Executed
1	0	0	0	NO
2	0	0	1	NO
3	0	1	0	NO
4	0	1	1	YES
5	1	0	0	NO
6	1	0	1	YES
7	1	1	0	NO
8	1	1	1	YES

Coverage Type	Minimum # of Test Cases	Possible Combinations
Statement	1	4 or 6 or 8

Verification – Testing examples (2/3)

```
IF(C AND( A OR B))
   THEN Foo();
```

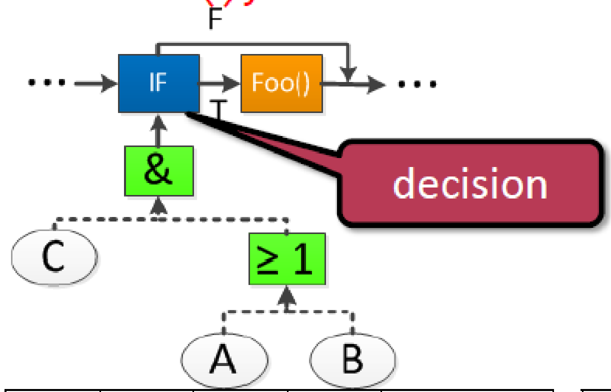


#	A	B	C	Foo Executed
1	0	0	0	NO
2	0	0	1	NO
3	0	1	0	NO
4	0	1	1	YES
5	1	0	0	NO
6	1	0	1	YES
7	1	1	0	NO
8	1	1	1	YES

- Decision Condition Coverage (DC) Level B
  - Each decision tries every possible outcome
  - Each entry and exit point is invoke

Verification – Testing examples (2/3)

```
IF(C AND( A OR B))
   THEN Foo();
```



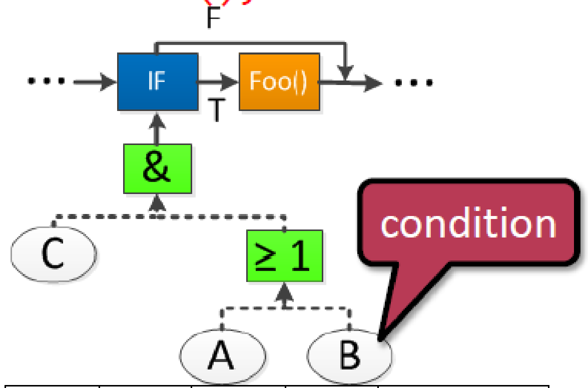
#	A	B	C	Foo Executed
1	0	0	0	NO
2	0	0	1	NO
3	0	1	0	NO
4	0	1	1	YES
5	1	0	0	NO
6	1	0	1	YES
7	1	1	0	NO
8	1	1	1	YES

- Decision Condition Coverage (DC) Level B
  - Each decision tries every possible outcome
  - Each entry and exit point is invoke

Coverage Type	Minimum # of Test Cases	Possible Combinations
Statement	1	4 or 6 or 8
Decision	2	4 or 6 or 8 + Any NO

Verification – Testing examples (3/3)

```
IF(C AND( A OR B))
   THEN Foo();
```

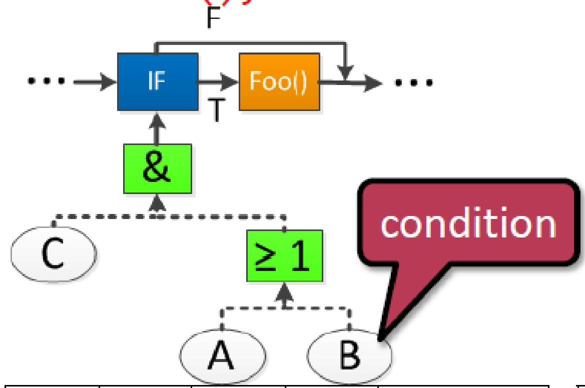


#	A	B	C	Foo Executed
1	0	0	0	NO
2	0	0	1	NO
3	0	1	0	NO
4	0	1	1	YES
5	1	0	0	NO
6	1	0	1	YES
7	1	1	0	NO
8	1	1	1	YES

- Modified Decision Condition Coverage (MCDC) Level A
  - Each decision tries every possible outcome
  - Each condition in a decision takes on every possible outcome
  - Each entry and exit point is invoked
  - Each condition in a decision is shown to independently affect the outcome of the decision

Verification – Testing examples (3/3)

```
IF(C AND( A OR B))
   THEN Foo();
```



#	A	B	C	Foo Executed
1	0	0	0	NO
2	0	0	1	NO
3	0	1	0	NO
4	0	1	1	YES
5	1	0	0	NO
6	1	0	1	YES
7	1	1	0	NO
8	1	1	1	YES

- Modified Decision Condition Coverage (MCDC) Level A
  - Each decision tries every possible outcome
  - Each condition in a decision takes on every possible outcome
  - Each entry and exit point is invoked
  - Each condition in a decision is shown to independently affect the outcome of the decision

Coverage Type	Minimum # of Test Cases	Possible Combinations
Statement	1	4 or 6 or 8
Decision	2	4 or 6 or 8 + Any NO
MCDC	4	2,3,4, and 6 OR 2,4,5 and 6

ANNEXES

DO-330 - Software Tool Qualification Considerations

- Tools can introduce errors into the final system.
  - Especialy, SW development tools.

- They are verified on the same level as the developed application.

- The DO-330 « Software Tool Qualification Considerations » specifies 3 qualified tools categories and 5 TQL.
  - Corresponding more or less to DAL.

- E.g., Scade Suite, Matlab Stateflow, Wind River Diab compiler

DO-331 - Model-Based Development and Verification Supplement to DO-178

- Use of models for source code synthesis and verification
- Early model based validation
- Matlab Simulink (already used), AADL

DO-332 - Object-Oriented Technology and Related Techniques Supplement

Oversees the OOP related techniques, including:

- parametric polymorphism
- Overloading
- type conversion
- exception management
- dynamic memory management
- Virtualization

Liskov Substitution Principle

DO-333 - Formal Methods Supplement

Already used in many projects

Mature technologies available

Defines how certification credits can be earned by its use

Can be part of the Development process

Typical tools:

- Model checker
- Static code analyzers
- Theorem provers (only in limited scenarios)

POWERED BY TRUST