DO-178C/ED-12C

The new software standard for the avionic industry: goals, changes and challenges

SVEN NORDHOFF
Aerospace Certification / Process Assurance & SPICE Assessor
sven.nordhoff@sqs.com

Sven Nordhoff holds a diploma in Computer Science and has been working for the SQS Market Unit Industrial Service & Solutions for twelve years. He is primarily responsible for Aerospace Certification and Process Assurance, which includes SW/HW monitoring of suppliers at Airbus Operations, Germany. He also is a Principal ISO-15504/SPICE Assessor and teaches seminars on process improvement, quality assurance and airworthiness standards. As a member of the international working group EUROCAE/RTCA WG71/SC205, he has been involved in the development of avionic standard DO-178C from day one.
1 MANAGEMENT SUMMARY

The standard DO-178C/ED-12C, ‘Software Considerations in Airborne Systems and Equipment Certification’, is the upcoming international standard jointly published by the RTCA and EUROCAE. This new standard will replace DO-178B/ED-12B to be the primary document by which the aviation certification authorities such as the Federal Aviation Administration (FAA, USA) and the European Aviation Safety Agency (EASA) approve all commercial software-based aerospace systems. DO-178B/ED-12B had been established in 1992 and it was necessary to update this standard to clarify some inconsistencies and introduce some new methodologies and technologies which have already been used in the current development and quality departments in the avionic industry. In addition, the new DO-178C/ED-12C has been established to ensure the validity of this standard for the future, in view of the fact that the old ‘B’ version has now been in use for over 20 years.

Essentially, this whitepaper summarises the following:

+ The goal and the methodology of this standard
+ The history and the activities which brought this standard to its current form
+ The main facts about DO-178C/ED-12C
+ The differences between DO-178B/ED-12B and DO-178C/ED-12C in general, and in particular regarding technological and methodological aspects
+ The impact of this new standard on development and quality departments all over the world
+ A short methodology and workflow how a company can ensure compliance to this standard
+ A way to avoid stumbling blocks and inconsistencies

The new standard DO-178C/ED-12C is divided into the core document, three supplements for the technology-specific parts (Model-Based Development & Verification, Object-Oriented Technology and Formal Methods), and a special document considering tool qualification. The key figures are identified and put into the context of the DO-178C ‘world of thinking’. The usage of this family of standards is explained and a possible workflow is suggested for the introduction of the new standard.

DO-178C/ED-12C is now officially finalised – the last plenary session was held in November 2011 in Daytona Beach, USA. All parts have been completed and the final step will be the formal approval of the RTCA (Radio Technical Commission for Aeronautics) and EUROCAE (a non-profit organisation providing a European forum for resolving technical problems with electronic equipment for air transport). The aviation authorities have been requested to determine whether DO-178C/ED-12C and its supporting documents can be considered ‘acceptable means’ for the certification of software-based systems. Once they have been approved by the authorities, these documents will apply to the next aircraft programmes, to future redesign of equipment, or to new equipment for existing aircraft or engines.

The author of this whitepaper has been a member of the DO-178C/ED-12C working group from the very beginning and leads a group of DO-178B/C specialists within SQS. The gap analysis methods presented below were established years ago for DO-178B and have been complemented to accommodate the new requirements of DO-178C. Many companies have defined or improved their DO-178 processes with the help of SQS.
Building aircraft is a rather important and challenging task, which requires a great amount of expert knowledge and companies with an enormous potential in terms of financial resources and strategic power. In recent years, the market for large commercial airplanes has been dominated by two major global players: Boeing and Airbus. In the future, more companies will enter this market, mainly encouraged to do so through political and financial support by their governments. Besides companies from Canada and Brazil, which have already joined the market, new companies from China and Russia are now starting to build large commercial airplanes.

The number of aircraft departures and flight hours has increased considerably over the last decades, and the number of aircraft is growing, too (see Figure 1). (I)
The future starts right here, with an explosion of aircraft orders: 797 orders for Boeing 787 (‘Dreamliner’), and 1,055 orders for Airbus A320 Neo. (2) These aircraft depend upon software-based embedded systems, which increases the necessity of quality assurance activities dramatically. Other companies from the new strong economies will enter the market as soon as possible to get a share of the large cake of selling commercial aircraft, but in order to be successful they need to consider the tremendous amount of quality measures which the aircraft authorities require.

If we look at the fatal accident rate of aircraft in the recent past, it can be observed that it is decreasing. Figure 2 shows the annual fatal accident rate. (1)

The increasing usage of commercial aircraft and the increasing complexity of the aircraft systems including software do not lead to a higher number of accidents and fatalities.

It seems that the introduction of process-related standards (not only for software), their consistent application within the industry, and the rigorous approval of software systems by the airworthiness authorities are reducing the number of failures caused by these highly integrated and complex systems. Apart from the strict introduction of quality-based components and greater experience with the structural behaviour of materials, the application of these process-related standards is the main success factor. There is no trend to decrease the level of regulation. The introduction of the new standard DO-178C/ED-12C will not weaken the ‘qualification’ activities but rather boost the enforcement of quality.

**Figure 2: Annual fatal accident rates for aircraft worldwide – 2010 Statistical Summary, June 2011**
3 HISTORY AND OVERVIEW OF AVIONIC STANDARDS

Standard DO-178B was established in 1992 as a successor of DO-178A (1985) and DO-178 (1980). The previous versions were often inconsistent in their wording and stood in the way of achieving the required goals. DO-178B offers a clear framework and methodology highly accepted by authorities, aircraft manufacturers and the supplier industry alike.

This success was achieved by eliminating the following aspects:
+ Product-specific requirements
+ Programming language and development method-specific features

Both DO-178B and its successor DO-178C concentrate on the following topics:
+ Focus on software by identifying interfaces only in terms of system and hardware aspects
+ Definition of criticality levels for software (SW level), derived from the associated ‘Failure Condition’
+ Definition of software life cycle processes and identification of quality criteria for each process, based on the specific SW level
+ Definition of required documents for each SW level, identifying an overall content structure
+ Focus on objectives, SW level applicability, and required outputs to ensure quality goals

Figure 3: Avionic standards for development purposes
DO-178B/C belong to a ‘family’ of similar standards which were established for the avionic industry for guidance:
+ To ensure safety processes and safety assessment (ARP 4761)
+ To ensure quality for complex systems (ARP 4754A)
+ To ensure quality for complex electronic hardware (DO-254)
+ To ensure quality for software systems (DO-178B and C)

All these standards are based on a clear philosophy:
+ To implement only the ‘intended’ functionality (and nothing else)
+ To be safety-driven, which means that for safety-critical application the processes are more rigorous
+ To implement a ‘V’-model approach for every development cycle (SW, HW and systems), whose components are dependent on each other and have clear interfaces

Figure 3 shows an overview of all standards which focus on development processes and quality criteria for the avionic industry.
All these standards share a major goal: the concentration on development processes and quality aspects specific to the required safety level.

This concept, on the one hand, ensures that necessary activities for safety-critical application are clearly specified and measures are defined to safeguard adequate implementation; on the other hand, for systems which are 'only' used for the comfort of the passenger, processes are less stringent.

Therefore, the quality criteria for safety-critical SW application within an aircraft, e.g. ‘Flight Controls’, are more rigorous than for uncritical software systems like ‘In-Flight Entertainment Systems’.

All these standards are based on a categorisation which defines failure conditions as shown in Figure 4 (see DO-178C, § 2.3.2, Table 2-1, p. 13). The DO-178C SW standard uses this kind of classification to define the objectives to be considered. These quality objectives are requirements which need to be proven to demonstrate compliance to DO-178C/ED-12C.

<table>
<thead>
<tr>
<th>SW LEVEL</th>
<th>FAILURE CONDITION</th>
<th>DESCRIPTION</th>
<th>WITH INDEPENDENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Catastrophic</td>
<td>71 (66)</td>
<td>30 (25)</td>
</tr>
<tr>
<td>B</td>
<td>Hazardous</td>
<td>69 (65)</td>
<td>18 (14)</td>
</tr>
<tr>
<td>C</td>
<td>Major</td>
<td>62 (57)</td>
<td>5 (2)</td>
</tr>
<tr>
<td>D</td>
<td>Minor</td>
<td>26 (28)</td>
<td>2 (2)</td>
</tr>
<tr>
<td>E</td>
<td>No Effect</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
</tbody>
</table>

Figure 5: Number of objectives for failure conditions

Figure 6: DO-178B/C concept regarding SW life cycle and processes
The number of quality objectives related to the SW level is identified in Figure 5 to show that the number of objectives is increasing the higher the safety level is. Quality objectives need to be addressed for the corresponding SW level in conjunction with the level of independence, meaning that at least one other person has to check the adequacy of this activity. The numbers in brackets refer to DO-178B.

‘No objective’ (DAL E) does not automatically mean that nothing is to be done. For example, Airbus Directives (ABD) require industry-conform SW engineering practices for SW level E.

As mentioned before, the avionic standards mainly deal with the development life cycle and processes. Therefore, DO-178C clearly defines an SW life cycle and processes which need to be followed.

There is no secret behind this concept nor are the chosen processes specific to the avionic industry. However, if we are looking into the details, some of the principles are remarkable:

- The usage of high-level requirements (in SW requirements processes) and low-level requirements (in SW design processes), which have to be tested (verified) adequately.
- The concept of ‘derived’ requirements (without traceability to the high-level requirements) which need to be analysed within the system safety activities to preclude that these requirements contradict the needs based on the safety classification.
- The usage of a SW ‘Planning Process’ and ‘Certification Process’ to establish an agreement with the authorities (e.g. EASA, FAA).

### 4 THE BEGINNING OF DO-178C/ED-12C

The ‘old’ B version of DO-178 merely focuses on defining life cycle processes and quality objectives to check the adequacy of these activities. Several supporting papers were generated over the years to clarify some aspects which were not specified clearly in DO-178B. These supporting papers were:

- Final Report for Clarification of DO-178B/ED-12B (3)
- Certification Authorities Software Team papers (4)
- JAA/EASA Certification Review Items (CRI, EASA, AIRBUS)
- Some FFA papers (issue papers, AC, AR, notices, OOTIA, research reports, etc.)
- EASA Memorandum for Software Aspects of Certification (5)

For the new version of the standard, the following topics were of special interest due to the fact that many development departments within the avionic industry are influenced by or want to use these technologies and related tools:

- Model-Based Development & Verification
- Object-Oriented Languages
- Commercial Off-The-Shelf Software (COTS)
- Formal Methods

All these aspects were drivers to start the DO-178C development. In March 2005, the first meeting was held in Washington DC, USA, with participants from EUROCAE Working Group 71 and RTCA Special Committee 205.
5 FACTS ON DO-178C/ED-12C

The new DO-178C/ED-12C family is structured as follows:

- **DO-178C/ED-12C Core Document**
- **DO-248C/ED-94C Supporting Information for ED-12C and ED-109A**
- **DO-xxx/ED-215 Tool Qualification Considerations**
- **DO-xxx/ED-218 Model-Based Development and Verification Supplement**
- **DO-xxx/ED-217 Object-Oriented Methods Supplement**
- **DO-xxx/ED-216 Formal Methods Supplement**

The new DO-178C/ED-12C family is structured as follows:

The DO-178C core document is the successor of DO-178B with the same structure and a similar approach. The main objective for this document was to be

+ only guidance material (with clear rules and objectives) and
+ technology- and methodology-independent.

The structure of the core document is very similar to DO-178B, but the following aspects are either new or have been changed:

+ Establishment of ‘Rationales’ for every objective of DO-178C/ED-12C. These rationales are listed in the DO-248C document. For example, the rationales for the objectives concerning structural ‘Code Coverage’ are:

`Table A-7 – Verification of Outputs of Software Testing: Objectives 5, 6, and 7 ensure that test cases written for requirements explore the source code with the degree of rigor required by the software level. For level C, it was deemed satisfactory to demonstrate that all statements in the source code were explored by the set of test cases. For level B, the addition of the requirement that all decision paths in the source are covered was considered sufficient to address the increase in the associated hazard category. However, for level A, the committee established that all logic expressions in the source code should be explored. The use of techniques such as multiple condition decision coverage, or exhaustive truth table evaluation to fully explore all of`
the logic was considered impractical [...] The compromise was achieved based on hardware logic testing that concentrated on showing that each term in a Boolean expression can be shown to affect the result. The term for this type of coverage was Modified Condition/Decision Coverage (MC/DC).'

+ Robustness aspects improved
+ User-modifiable SW aspects improved
+ ‘Testing vs. Data and Control Coupling’ improved
+ Guidance to auto-code generator added
+ Untraceable code added
+ ‘Parameter Data Item’ consideration added

But the main improvement within the new DO-178C/ED-12C is the establishment of so-called ‘Supplements’ providing technology- and method-specific material which required a more detailed and restrictive mapping with regard to DO-178C/ED-12C.

The following supplements were generated.

+ Software Tool Qualification Considerations (DO-xxx/ED-215)
+ Model-Based Development & Verification Supplement (DO-xxx/ED-218)
+ Object-Oriented Technology Supplement (DO-xxx/ED-217)
+ Formal Methods Supplement (DO-xxx/ED-216)

The RTCA has not yet assigned DO numbers to these supplements. Strictly speaking, the Software Tool Qualification Considerations document is not just a supplement, because its use is also intended for other industry domains. For the purpose of the present whitepaper, however, it shall be treated as a supplement.

DO-248/ED-94C is a guideline document containing additional supporting and explanatory material, including:
+ Frequently Asked Questions (FAQs)
+ Discussion Papers
+ The DO-178C/ED-12C Rationales
+ Difference between DO-178C/ED-12C and DO-178B/ED-12B

DO-278A/ED-109A is the guidance document for CNS/ATM systems (Software Integrity Assurance Considerations For Communication, Navigation, Surveillance and Air Traffic Management Systems). Its structure is very similar to that of DO-178C, showing the same approach but with an emphasis on Commercial Off-The-Shelf SW (COTS).

The following sections give a short overview of the core document and its supplements. The details of DO-248/ED-94C and DO-278A/ED-109A shall not be considered in this whitepaper.

5.1 GENERAL OVERVIEW OF THE DO-178C/ED-12C CORE DOCUMENT

The main content of the DO-178C/ED-12C core document is the definition of SW life cycle processes and related activities. Among these activities, the following are the most important:

+ Planning Process
  · Establishment of SW Plans
  · Definition of the SW Life Cycle Environment
  · Language and Compiler Considerations
  · Establishment of SW Standards
  · Review and Assurance of the SW Planning Process
+ Requirements Process
  · Development of High-Level Requirements
  · Development of ‘Derived’ High-Level Requirements
+ Design Process
· Development of SW Architecture
· Development of Low-Level Requirements
· Development of ‘Derived’ Low-Level Requirements
· Considerations for User-Modifiable Software and Deactivated Code
+ Coding Process
· Development of Source Code
+ Integration Process
· Executable Object Code is Loaded into Target Hardware for HW/SW Integration
+ Verification Process
· Reviews and Analyses of High-Level Requirements
· Reviews and Analyses of Low-Level Requirements
· Reviews and Analyses of Software Architecture
· Reviews and Analyses of Source Code
· Reviews and Analyses of the Outputs of the Integration Process
· Hardware/Software Integration Testing
· Software Integration Testing
· Low-Level Testing
· Requirements-Based Test Coverage Analysis
· Structural Coverage Analysis
· Reviews and Analyses of Test Cases, Procedures and Results
· Software Development Process Traceability
· Software Verification Process Traceability
· Verification of Parameter Data Items
+ Configuration Management Process
· Configuration Identification
· Baselines and Traceability
· Problem Reporting, Tracking and Corrective Action
· Change Control
· Change Review
· Configuration Status Accounting
· Archive, Retrieval and Release
· Data Control Categories
· Software Load Control
· Software Life Cycle Environment Control
+ Quality Assurance Process
· Quality Assurance Activities
· Software Conformity Review
+ Certification Liaison Process
· Means of Compliance and Planning
· Compliance Substantiation

For every software, level-specific life cycle documents are needed. In § 11.0, detailed requirements including naming and structure are listed. Figure 8 describes the names, objectives and related control categories of the documents which indicate the rigour of the configuration- and change management.

Additional Considerations (§ 12.0) deal with objectives and/or activities which may replace, modify, or add objectives and/or activities defined in the rest of DO-178C/ED-12C:
+ Use of Previously Developed Software
+ Tool Qualification
+ Alternative Methods
· Exhaustive Input Testing
· Multiple-Version Dissimilar Software Verification
· Software Reliability Models
· Product Service History

The most important section in DO-178C/ED-12C is Annex A (Process Objectives and Outputs by Software Level), where the following aspects are defined for every process group:
+ Objectives
+ Applicability by SW Level
+ Output Documents
+ Control Category
+ Independence
<table>
<thead>
<tr>
<th>SOFTWARE LIFE CYCLE DATA</th>
<th>OBJECTIVE</th>
<th>§</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSAC - Plan for Software Aspects of</td>
<td>Plan to describe the means of compliance for certification-relevant aspects</td>
<td>11.1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>Certification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDP - SW Development Plan</td>
<td>Plan to describe the development process and standards</td>
<td>11.2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SVP - SW Verification Plan</td>
<td>Plan to describe all verification activities</td>
<td>11.3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SCMP - SW Configuration Management Plan</td>
<td>Plan to describe the configuration management processes</td>
<td>11.4</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SQAP - SW Quality Assurance Plan</td>
<td>Plan to describe the quality assurance processes</td>
<td>11.5</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SRS - SW Requirements Standards</td>
<td>Standard for high-level requirements definition</td>
<td>11.6</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>SDS - SW Design Standards</td>
<td>Standard for SW architecture and low-level requirements definition</td>
<td>11.7</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>SCS - SW Coding Standards</td>
<td>Standard for coding</td>
<td>11.8</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>SRD - SW Requirements Document</td>
<td>High-level requirements</td>
<td>11.9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>SDD - SW Design Description</td>
<td>SW architecture and low-level requirements</td>
<td>11.10</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SRC - Source Code</td>
<td>Coding</td>
<td>11.11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>EXE - Executable Object Code</td>
<td>Executable file</td>
<td>11.12</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>SVCP - SW Verification Cases and Procedures</td>
<td>Document to identify verification, test cases and procedures</td>
<td>11.13</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SVR - SW Verification Results</td>
<td>Document to identify all verification and test results</td>
<td>11.14</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>PR - Problem Reports</td>
<td>List of all deviations</td>
<td>11.17</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SCMR - SW Configuration Management Record</td>
<td>Evidence about configuration management</td>
<td>11.18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SQAR - SW Quality Assurance Record</td>
<td>Evidence about quality assurance</td>
<td>11.19</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SCII - SW Environment Life Cycle Index</td>
<td>Environment definition and configuration control</td>
<td>11.15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>SAS - SW Accomplishment Summary</td>
<td>Document to show compliance substantiation to the authorities</td>
<td>11.20</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>SCI - SW Configuration Index</td>
<td>Document to clearly identify the SW and documentation configuration</td>
<td>11.16</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Control Category 1 includes all Configuration Management Activities defined by DO-178C.

Control Category 2 includes only a subset of Configuration Management Activities.

Figure 8: Life cycle documents and control category
<table>
<thead>
<tr>
<th>TABLE</th>
<th>PROCESS GROUP</th>
<th>NO. OF OBJECTIVES</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-1</td>
<td>Software Planning Process</td>
<td>7</td>
</tr>
<tr>
<td>A-2</td>
<td>Software Development Process</td>
<td>7</td>
</tr>
<tr>
<td>A-3</td>
<td>Verification of Outputs of Software Requirements Process</td>
<td>7</td>
</tr>
<tr>
<td>A-4</td>
<td>Verification of Outputs of Software Design Process</td>
<td>13</td>
</tr>
<tr>
<td>A-5</td>
<td>Verification of Outputs of Software Coding &amp; Integration Processes</td>
<td>9</td>
</tr>
<tr>
<td>A-6</td>
<td>Testing of Outputs of Integration Process</td>
<td>5</td>
</tr>
<tr>
<td>A-7</td>
<td>Verification of Outputs of Software Testing</td>
<td>9</td>
</tr>
<tr>
<td>A-8</td>
<td>Software Configuration Management Process</td>
<td>6</td>
</tr>
<tr>
<td>A-9</td>
<td>Software Quality Assurance Process</td>
<td>5</td>
</tr>
<tr>
<td>A-10</td>
<td>Certification Liaison Process</td>
<td>3</td>
</tr>
</tbody>
</table>

Figure 9: List of DO-178C/ED-12C process tables with objectives

<table>
<thead>
<tr>
<th>OBJECTIVE</th>
<th>APPLICABILITY FOR SW LEVEL</th>
<th>OUTPUT / DATA ITEM</th>
<th>CONTROL CATEGORY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Test procedures are correct.</td>
<td>● ○ ○</td>
<td>Software</td>
<td>2 2 2</td>
</tr>
<tr>
<td>2 Test results are correct and discrepancies explained.</td>
<td>● ○ ○</td>
<td>Software</td>
<td>2 2 2</td>
</tr>
<tr>
<td>3 Test coverage of high-level requirements is achieved.</td>
<td>● ○ ○ ○</td>
<td>Software</td>
<td>2 2 2 2</td>
</tr>
<tr>
<td>4 Test coverage of low-level requirements is achieved.</td>
<td>● ○ ○</td>
<td>Software</td>
<td>2 2 2</td>
</tr>
<tr>
<td>5 Test coverage of software structure (modified condition/decision coverage) is achieved.</td>
<td>●</td>
<td>Software</td>
<td>2</td>
</tr>
<tr>
<td>6 Test coverage of software structure (decision coverage) is achieved.</td>
<td>●</td>
<td>Software</td>
<td>2 2</td>
</tr>
<tr>
<td>7 Test coverage of software structure (statement coverage) is achieved.</td>
<td>● ○ ○</td>
<td>Software</td>
<td>2 2</td>
</tr>
<tr>
<td>8 Test coverage of software structure (data coupling and control coupling) is achieved.</td>
<td>● ○ ○</td>
<td>Software</td>
<td>2 2</td>
</tr>
<tr>
<td>9 Verification of additional code, that cannot be traced to source code, is achieved.</td>
<td>●</td>
<td>Software</td>
<td>2</td>
</tr>
</tbody>
</table>

Figure 10: DO-178C/ED-12C test process table with objectives
5.2 SOFTWARE TOOL QUALIFICATION CONSIDERATIONS

The purpose of this document is to provide software tool qualification guidance to help the tool vendor and/or user define the required activities. Additional information is provided in the form of FAQs.

Software tools are widely used to assist in developing, transforming, testing, analysing, producing, and modifying aircraft-based software programmes, their data, or their documentation. And in this context, tools that are used to eliminate, reduce, or automate a specific DO-178C/ED-12C software life cycle process without verification of the tool output, need particular qualification.

There are five Tool Qualification Levels (TQLs), which are used similarly to the SW level in the DO-178C/ED-12C core document. The kind of tool is defined by using three criteria to clarify the amount of qualification activities to be conducted for a particular tool:

<table>
<thead>
<tr>
<th>CRITERIA</th>
<th>EXPLANATION</th>
<th>KIND OF TOOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A tool whose output is part of the airborne software and thus could insert an error.</td>
<td>Development Tool</td>
</tr>
<tr>
<td>2</td>
<td>A tool that automates verification process(es) and thus could fail to detect an error, and whose output is used to justify the elimination or reduction of 1. Verification process(es) other than those automated by the tool, or 2. Development process(es) that could have an impact on the airborne software.</td>
<td>Verification Tool to verify the output of a verification or development tool</td>
</tr>
<tr>
<td>3</td>
<td>A tool that, within the scope of its intended use, could fail to detect an error.</td>
<td>Verification Tool</td>
</tr>
</tbody>
</table>

Figure 11: Tool criteria definition

<table>
<thead>
<tr>
<th>SW LEVEL</th>
<th>CRITERIA</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>TQL-1</td>
</tr>
<tr>
<td>B</td>
<td>TQL-2</td>
</tr>
<tr>
<td>C</td>
<td>TQL-3</td>
</tr>
<tr>
<td>D</td>
<td>TQL-4</td>
</tr>
</tbody>
</table>

Figure 12: Tool Qualification Levels (TQLs) and related SW level
Accordingly, the rigour of tool qualification can be defined by the criteria and the SW level of the operational SW for which the tool will be used.

The structure of this document is very similar to the DO-178C/ED-12C core document. All tool-relevant processes are defined and objectives and activities are listed depending on the Tool Qualification Level (TQL). A generic tool development process is established. Moreover, the ‘Objective Tables’ known from the DO-178C/ED-12C core document are established for each process, though here they are used for the related tool processes.

A rather interesting point is the differentiation between the tool developer and the tool user when a COTS tool is used. Both parties need to conduct their own qualification activities. The task of the tool user is limited to planning and integration activities regarding the operational environment, while the tool developer needs to achieve complete compliance to DO-178C/ED-12C. This generates a degree of effort at TQL1 which is similar to SW level A activities.

5.3 MODEL-BASED DEVELOPMENT & VERIFICATION SUPPLEMENT

This supplement deals with Model-Based Development & Verification (MBD&V) and was written to add, modify, and substitute the objectives defined in DO-178C/ED-12C. Essentially, the models are used
+ To develop an unambiguous expression of requirements and architecture;
+ To assist in automated code generation;
+ To assist in automated test generation;
+ As analysis tools for the verification of requirements and architecture; and
+ In simulations for the partial verification of requirements, architecture, and/or an executable object code.

The structure of this supplement is very similar to the DO-178C/ED-12C core document. The supplement adds model-based development aspects to DO-178C/ED-12C and expands chapters affected by MBD&V. Chapters of DO-178C/ED-12C which are not affected by MBD&V remain unchanged. From a DO-178C point of view, these models represent software requirements and/or architecture to support the software development and verification processes.

For every model, requirements need to be identified from which the model is developed. Those requirements should be external to the model and be a complete set of requirements and set of constraints (see MBD&V Supplement, § MB.1.6.1, p. 2). The supplement deals with two types of models:
+ Specification Models containing high-level requirements
+ Design Models containing architecture and low-level requirements

Figures 13 and 14 demonstrate the different usages of models (Light green) within the context of DO-178C/ED-12C processes (see MBD&V Supplement, Table MBJ.1-1, p. 4).

The following aspects described in the MBD&V Supplement should be highlighted:
+ In the Planning Phase and in the planning documentation, the usage of models needs to be explained and the model needs to be categorised as a specification or design model. All MBD&V methods used need to be identified during the planning phase, including verification methods like model coverage analysis criteria.
+ SW Model Standards are required.
+ Simulation can be used in design models to support testing and analysis methods. The usage of simulation in a model may produce
simulation cases, procedures, and results which need to be verified (MB.C-3 objectives MB8–MB10: Requirements; MB.C-4 objectives MB14–MB16: Design; and MB.C-7 objectives MB10–MB12: Verification).

+ **Model Coverage Analysis** supports the detection of unintended functions in the design model by determining which requirements expressed by the design model were not exercised, through verification based on the requirements from which the model was developed.

+ **Usage of Model Simulation** for
  · Verification of the model, and
  · Verification of the executable object code.

<table>
<thead>
<tr>
<th>DO-178C PROCESSES</th>
<th>EXAMPLE 1</th>
<th>EXAMPLE 2</th>
<th>EXAMPLE 3</th>
<th>EXAMPLE 4</th>
<th>EXAMPLE 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Requirements and System Design Process</td>
<td>Requirements allocated to SW</td>
<td>Requirements from which the model is developed</td>
<td>Requirements from which the model is developed</td>
<td>Requirements from which the model is developed</td>
<td>Requirements from which the model is developed</td>
</tr>
<tr>
<td>SW Requirements and SW Design Process</td>
<td>Requirements from which the model is developed</td>
<td>Specification Model</td>
<td>Specification Model</td>
<td>Design Model</td>
<td></td>
</tr>
<tr>
<td>Software Coding Process</td>
<td>Design Model</td>
<td>Design Model</td>
<td>Textual Descriptions</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 13:** Examples of usage of specification and design models

**Figure 14:** Model coverage criteria to identify unintended functionality
Clarification of the model coverage criteria is of crucial importance to identifying unintended functions. Figure 14 shows some examples of criteria which are recommended by the supplement (see MBD&V Supplement, Table MB.6-1, p. 32).

The last chapter of the supplement gives some examples clarifying the usage of the supplement with regard to the relationship between a design model or specification model and DO-178C high-level requirements, low-level requirements, and software architecture.

5.4 OBJECT-ORIENTED TECHNOLOGY SUPPLEMENT

Object-Oriented Technologies (OOT) and programming languages like Java and C++ have been widely used for decades in the commercial/industrial sectors where safety is not critical. In the avionic industry, the usage of OOT is increasing but the issues with regard to certification did pose a serious problem because no proper guidelines were available to clarify usage and certification. For example, for SW level A, B and C software, code coverage measures need to be taken (DO-178C/ED-12C is talking about Structural Coverage, SW Level A → MC/DC Coverage, SW Level B → Decision Coverage, SW Level C → Statement Coverage). This DO-178C/ED-12C objective is very clear for the classical functional programming languages like ‘C’, but for object-oriented languages, e.g. with encapsulation, polymorphism and overloading, the meaning of ‘Code Coverage’ is quite different.

The supplement ‘Object-Oriented Technology and Related Techniques (OOT)’ starts with Chapter OO.1.0 outlining the characteristics of these techniques. The supplement was written to be programming-language independent, using definitions which are understood without any specific language background. This became necessary because the terminology of the different OOT and programming language approaches usually is quite different, which regularly results in Babylonian situations.

The next chapters of the OOT Supplement are structured similarly to the DO-178C/ED-12C core document, expanded by adding, modifying, or deleting DO-178C/ED-12C objectives relating to OOT.

The following aspects of the OOT Supplement should be highlighted:

+ In the Planning Phase and in the planning documentation, the usage of OOT is to be explained. All virtualisation techniques used and any reuse of components need to be identified during the planning phase, including all verification methods employed to achieve the objectives of DO-178C/ED-12C and its supplements.

+ The scope of Verification activities must be widened to verify, e.g., the class hierarchy, local type consistency, memory and exception management.

The Annex OO.D – Vulnerability Analysis deals with all the features of OOT, discussing the specific vulnerabilities and related guidance. In addition, supporting information (guidelines) is listed in this chapter to help identify possible solutions and clarify the advantages and disadvantages of the various methods.

Figure 15 shows some examples of possible solutions suggested by the supplement to cope with OOT-specific problems.
## 5.5 FORMAL METHODS SUPPLEMENT

The Formal Methods Supplement deals with formal methods to be used for avionic projects. Formal methods are mathematically based techniques for the
+ Specification,
+ Development, and
+ Verification
of software aspects of digital systems (see ED-216, § FM1.0, p. 1). The mathematical basis of formal methods consists of formal logic, discrete mathematics, and computer-readable languages. The use of formal methods is motivated by the expectation that performing appropriate mathematical analyses can contribute to establish the correctness and robustness of a design.

The following characteristics are associated with formal methods:
+ Unambiguously describing requirements of software systems
+ Enabling precise communication between engineers
+ Providing verification evidence such as consistency and accuracy of a formally specified representation of software
+ Providing verification evidence of the compliance of one formally specified representation with another

Section FM.1.0 contains explanatory text to aid the reader in understanding formal methods, and therefore is not to be taken as guidance.

### OOT ISSUES | PROPOSED PROCEDURES/METHODS
--- | ---
Dynamic Memory Management | Object pooling
 | Stack allocation
 | Scope allocation
 | Manual heap allocation
 | Automatic heap allocation

**Structural Coverage**

An acceptable means for demonstrating type consistency is by showing the software satisfies the Liskov Substitution Principle (LSP) (see ED-217, § OO.1.6.1.2.1, Liskov Substitution Principle, p. 4). This may be shown through test or formal methods.

1. Execute the requirements-based tests to capture the data for structural coverage analysis.
2. If type consistency is shown, then evaluate at least one instance of one of the classes that can occur at each call point.
3. If the type consistency cannot be satisfied, then evaluate at least one instance of each class that can occur at each call point (pessimistic).
4. Perform structural coverage analysis for the appropriate level, include data and control flow analysis.
5. Consider all inherited as well as explicit methods for each class.

Figure 15: OOT issues and proposed solutions
Annex FM.A of this document describes how the DO-178C/ED-12C objectives are revised in line with this formal methods guidance.

Establishing a formal model of the software artefact is fundamental to all formal methods. In general, a model is an abstract representation of a given set of aspects of the software that is used for analysis, simulation, and/or code generation. A model should have an unambiguous, mathematically defined syntax and semantics. This makes it possible to use automated means to obtain guarantees that the model has certain specified properties.

The chapters of the FM Supplement are structured similarly to the DO-178C/ED-12C core document, and expanded by adding, modifying, or deleting DO-178C/ED-12C objectives relating to FM. Chapters of DO-178C/ED-12C which are not affected by formal methods remain unchanged.

The following aspects of the FM Supplement should be highlighted:
+ In the planning phase and in the planning documentation, the usage of FM must be explained.
+ Requirements or design artefacts can be defined with the help of formal models. This allows for some of the verification objectives to be satisfied by the use of formal analysis.
+ With formal analysis, the correctness of life cycle data with respect to a formal model can be proved or disproved.
+ If formal methods are used for verification purposes, the following considerations are important:
  - All notations used for formal analysis should be verified to have a precise, unambiguous, mathematically defined syntax and semantics.
  - The soundness of each formal analysis method should be justified. A sound method never asserts that a property is true when it may not be true.
  - All assumptions related to each formal analysis should be described and justified; such as, for example, assumptions associated with the target computer or about the data range limits.
+ Reviews and analyses of the formal analysis cases, procedures, and results for requirements, architecture, and source code are necessary.

For the verification of SW requirements/SW design, the following additional objectives are defined:
+ Formal analysis cases and procedures are correct (FM8, FM14).
+ Formal analysis results are correct and discrepancies explained (FM9, FM15).
+ Requirements formalisation is correct (FM10, FM16).
+ The formal method is correctly defined, justified, and appropriate (FM11, FM17).

For the verification of SW testing, the following additional objectives are defined:
+ Coverage of high-level requirements is achieved (FM3).
+ Coverage of low-level requirements is achieved (FM4).
+ Complete coverage of each requirement is achieved (FM5).
+ The set of requirements is complete (FM6).
+ Unintended dataflow relationships are detected (FM7).
+ Dead code and deactivated code are detected (FM8).
6 GAP ANALYSIS AND WAY OF WORKING WITH DO-178C

In this chapter, a method is described how to reach compliance to DO-178C and to check the necessity of applying the DO-178C supplements. This so-called SQS gap analysis starts with the identification of the current maturity level of a project with regard to DO-178C, and the way to achieve final compliance to this standard. Depending on the given maturity level, the time required to reach compliance will be shorter or longer. If the maturity level is low, it is advisable to introduce an expert to the project so that improvement can be started with guided support.

The following maturity levels can be identified:

+ **Maturity Level 3**
  DO-178B processes were successfully introduced before.

+ **Maturity Level 2**
  Safety-related process-driven SW development was introduced, but DO-178B has not been introduced.

+ **Maturity Level 1**
  Process-driven SW development was introduced, but DO-178B or safety-critical development has not been introduced.

+ **Maturity Level 0**
  No SW processes established to date.

Maturity Levels 3 and 2 are good starting points to achieve compliance to DO-178C/ED-12C in the short term.

With Maturity Level 3, the earlier usage of DO-178B-compliant processes only requires a minimal amount of gap analysis in order to identify the differences between DO-178B and C (see the previous chapter for details). No new processes or changes in methodologies need to be addressed, which results in manageable efforts.

If your SW processes are driven by embedded and safety-critical application development, like IEC 62304 (medical) or IEC 61508 (safety-critical industry), the steps toward DO-178B and C are not too big (Maturity Level 2). But some methodology changes may be necessary, which would result in medium investments. For example, the so-called low-level requirements needed for DO-178C compliance are not required by other standards like IEC 62304. Therefore, the requirements structure and the test approach need only be adjusted slightly.

If, on the other hand, you are using industry-related processes with only black-box testing and without clear requirements management and quality assurance activities (Maturity Level 1), quite a number of aspects have to be introduced to be compliant to DO-178B/C. The analysis needs to start with the identification of all the processes used. Then the missing processes must be integrated into the existing process landscape. The final step is compliance to all DO-178C objectives, which results in more activities or a more stringent application of existing activities within the different processes.

If a company starts from scratch, without having either a SW process structure or experience with the safety-critical development of software, the management need to be aware of the fact that a lot of time, a huge amount of investments, and the attendance of DO-178C experts will be necessary to reach the aim. With the right spirit though, it is not impossible.

The best way to ensure compliance is using a gap analysis with the help of technical and DO-178-related expertise and employing a sound strategy to
identify required process steps and major inconsistencies. It is necessary to establish a common process framework with a set of standardised templates for all related DO-178B/C SW plans and documents.

The fulfilment of DO-178C/ED-12C-related processes can be achieved manually, but it will be easier to build up a set of tools to support the development, verification and test activities. Before a tool is acquired, it is necessary to check whether the tool (and the vendor) is compliant to the rules of DO-178C/ED-12C. With most of the activities, the user has to ensure the tool qualification – the vendor can support the activities but cannot solve all the problems alone.

The process workflow in Figure 16 provides a short overview of which activities are necessary to ensure compliance to DO-178C/ED-12C objectives.

In general, this workflow should ensure compliance to the DO-178C/ED-12C core document.

As the supplements extend the guidance from DO-178C/ED-12C for a specific technology, the gap analysis should first consider all of the standard’s objectives, and then the supplements’ objectives. After that, any additional considerations may follow.

Figure 16: SQS workflow for DO-178C/ED-12C gap analysis
Figure 17: SQS workflow for DO-178C/ED-12C/Supplement introduction
The best approach is to consolidate all the objectives of DO-178C/ED-12C and the applicable supplements. For each objective, it is necessary to make a statement on how compliance will be achieved, along with identifying any applicable life cycle data items. Because the objective numbering scheme of the Annex A Tables of DO-178C/ED-12C and the Annex A Tables of the supplements is unique, the identification of the objectives and their document sources will be clear and unambiguous.

As mentioned above, the supplements are subdivided into guidance and guideline material. The guidelines include introductory material and FAQs. The OOT Supplement, for instance, clarifies some aspects by using a vulnerability analysis, the MBD&V Supplement describes examples of models, and the Formal Methods Supplement gives advice on which activities formal methods can be used for.

Figure 17 shows a general workflow used by SQS, considering the integration of the different supplements within the gap analysis.

The most important aspect when using this workflow is to check which guidance needs to be addressed with regard to the different supplements. Various examples presented in the supplements show the usage of the guidelines and their adequate interpretation.

7 CONCLUSION AND OUTLOOK

The predecessor of DO-178C – standard DO-178B – had been one of the most successful international software standards ever. In its time (1990–2011), the complexity of aircraft systems increased manifold whereas the number of aircraft accidents related to software failures declined. This most certainly is a huge success story.

During the validity of DO-178B, i.a. the aircraft types Airbus A330/340, Airbus A380, Airbus A400M and Boeing 787 conquered the market boasting software systems of previously unknown complexity. At the same time, new SW development technologies and methodologies were introduced that did not have a clear and commonly agreed certification basis.

The committee working on DO-178C was composed of authorities, aircraft manufacturers, system suppliers, tool vendors and consultants with experience from all fields necessary to develop such a standard. In the end, the new DO-178C/ED-12C standard with its four supplements filled over 650 pages, six times the volume of DO-178B.

The DO-178C/ED-12C core document is not revolutionary - it is a slight improvement in comprehensibility and usability, clearly separating guidance from guidelines. The most important improvement is the establishment of the four supplements in order to provide more guidance for the interesting technology-specific fields such as Model-Based Development & Verification and Object-Oriented Methods.

The document covering Tool Qualification was needed because the introduction of an increasing number of tools for development and verification assistance requires a lot of guidance in this field. Moreover, the usage of verification tools qual-
ifying the output of a development tool had not been within the scope of DO-178B, and it was time to consider this type of tool.

Object-oriented languages are very well known in commercial software development, but in the avionic industry OOT is employed mainly at the requirements and design level, for example using UML for a more descriptive representation. Object-oriented languages have not been used at all for the most safety-critical SW levels A–C due to the fact that compliance to the objectives of DO-178B is not easily achieved. Additional supporting papers (6) were generated over the last ten years to close this gap, but without success. With DO-178C and the OOT Supplement, guidance now is straightforward, and all companies using OOT are able to check their compliance.

Model-Based Development & Verification is well established in the field of avionic projects. This is mainly due to the fact that there are some tools on the market which already qualified against DO-178B. Also, with ‘Software Code Generation’ the industry can rely on proven concepts which have already been used for years for the most critical software parts within an aircraft (e.g. Flight Controls in Airbus aircraft). Nevertheless, many issues like model coverage and simulation were raised by the authorities and aircraft manufacturers, which were then adequately addressed in the MBD&V Supplement.

In the context of standards, the crucial task is to achieve compliance: the above Maturity Level concept supports this effort and identifies activities and measures that are necessary to ensure this aim. In addition, a workflow explains how to consider one or more supplements to introduce new methodologies and technologies. Structurally, these supplements are similar to the DO-178C core document but in fact they add, modify, or delete DO-178C/ED-12C objectives. Companies which already use the methodologies and technologies addressed by the supplements are invited to show compliance as soon as possible. If they fail to do so, they run the risk of non-compliance in case they have to re-certify their projects. This aspect is the most challenging task of the new DO-178C/ED-12, and all companies must address these issues at their earliest convenience. DO-178C/ED-12C experts and quality assurance specialists can support them and help them reach these goals in time and within budget.


4. CAST. Certification Authorities Software Team. http://www.faa.gov/aircraft/air_cert/design_approvals/air_software/cast/cast_papers/: FAA.

