Modification to Legacy Software Developed per DO-178A Level 1 to DO-178B Level A: How to Organize Software Life Cycle Data for Software Approval in Aircraft Certification

Johnny Cardoso Marques
EMBRAER – Av. Brigadeiro Faria Lima 2170 – São José dos Campos – SP – Brazil
FACULDADE ANHANGUERA – Av. Charles Schneider, 585 – Taubaté – SP – Brazil
johnny.marques@embraer.com.br

Abstract

A robust and well-organized process reflects the maturity of the development team, which is one of the key points for developing critical software safety. This is also includes the use of Legacy Software. The implications of Legacy Software in a certification process is commonly misunderstood by the software developers. This paper presents a guideline to apply modifications using DO-178B level A objectives to previously developed and approved software per DO-178A Level 1.

1. Introduction

The purpose of certification activity in civil aviation is to establish, through requirements, a minimum level of safety that must be checked in every aircraft or product, and technically supervise the implementation of those requirements. [1][2]

The certification must cover the design, manufacture, operation and maintenance of aeronautical products. It is considered aeronautical product every aircraft, engine or propeller, and every component or part, which is part of the same aircraft, engine or propeller. [3] An applicant, is an aircraft manufacturer and is responsible for obtaining the approval of a type design of an aeronautical product. [3][4]

For the development of aeronautical systems there are a number of recommendations, formatted rules that describe the entire process life cycle of a system. This includes the process of developing software and hardware, significant processes to establish the safety of the system and details about features, requirements, and implementation. These recommendations or rules are the result of the harmonization between the aviation community, which includes authorities, manufacturers and suppliers. [4][5]

The aircraft certification requirements do not cover the software directly. Some documents from outside entities such as the Radio Technical Commission for Aeronautics (RTCA), guide the development of quality software for the industry, by the use of DO-178 standard. The certification authorities recognize that DO-178B as a valid mean to secure approval of embedded software. [6][7]

2. DO-178 history and legacy software

DO-178 [9] was developed to establish considerations for developers, installers, and users when designing an embedded equipment using software. Nowadays, DO-178 is in revision B (DO-178B [8]). DO-178 was originally published in 1982 and the current revision is dated 1992. In 1985, the revision A was established (DO-178A [10]). Since 2005, RTCA is promoting the special committee SC-205 to review DO-178B and create the DO-178C, that is expected to be released in the end of 2011. Table 1 establishes the evolution and main modifications between revisions.

The use of previously developed and approved software using superseded versions of DO-178 is a concern for the certification authorities, especially in a new aircraft project. The certification authorities classify the previously approved software using DO-178 or DO-178A using the terminology Legacy Software.

The FAA issued in 2004 the Order 8110.49 to address some subjects of concern in software approval. The chapters 10 and 11 of Order 8110.49 are
applicable for the use of Legacy Software, when a modification in previously software developed per DO-178 or DO-178A will be conducted. [13]

This paper will only produce results to be used in software previously approved using DO-178B, based in legacy software developed per DO-178A. The use of DO-178, in original version, is out of the scope of this paper.

Table 1. DO-178 evolution [11]

<table>
<thead>
<tr>
<th>Standard</th>
<th>Main Modifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO-178</td>
<td>• First release</td>
</tr>
<tr>
<td></td>
<td>• Not much useful information</td>
</tr>
<tr>
<td>DO-178A</td>
<td>• Modern software engineering principles</td>
</tr>
<tr>
<td></td>
<td>• Focus on documentation</td>
</tr>
<tr>
<td></td>
<td>• Poor verification practices</td>
</tr>
<tr>
<td></td>
<td>• 3 Software Levels</td>
</tr>
<tr>
<td>DO-178B</td>
<td>• A “what” not a “how” document.</td>
</tr>
<tr>
<td></td>
<td>• Focus on objectives and software life cycle data</td>
</tr>
<tr>
<td></td>
<td>• 5 Software Levels</td>
</tr>
<tr>
<td></td>
<td>• Tool qualification included</td>
</tr>
</tbody>
</table>

Accordingly FAA Order 8110.49, Chapter 10, after performing software change impact analysis, if modification is classified as Major, DO-178B must be used to address the software modification. Figure 1 provides a graphical representation on how it works. A modification can be classified as major or minor, accordingly FAA Order 8110.49, Chapter 10. This document defines the criteria to classify a modification. [13]

DO-178A specifies only three software levels. Among the three software levels (1, 2 and 3), level 1 is the most rigorous, and requires more documents to be generated and available to certification authority. [10]

DO-178B specifies five software levels. Each level of software has an accuracy of different development process and a set of objectives that must be met to approve the embedded software as part of an aircraft certification. Among the five software levels (A, B, C, D and E), level A is the most rigorous. [9]

According to [1], each system failure should be classified as an associated criticality and this classification is also made in five types, as described in Table 2.

A System Failure Hazardous Analysis (SFHA) is required to determine the contribution of software to potential failure conditions. The SFHA will identify the most critical failure condition, and this will drive the required software level.

After the determination of required software level, the aircraft manufacturer will perform a trade off with system suppliers to determine if a new system developed is required or if a reuse from a previously developed and certified system is possible.

This paper focus on the use of Legacy Software previously developed and certified per DO-178A Level 1 in other aircraft and will be modified as a modification using DO-178B Level A objectives for a new aircraft development and certification.

Table 2. Correlation between critical failure condition and software level

<table>
<thead>
<tr>
<th>System Most Critical Failure Condition/Software Contribution</th>
<th>Required Software Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catastrophic</td>
<td>A</td>
</tr>
<tr>
<td>Hazardous</td>
<td>B</td>
</tr>
<tr>
<td>Major</td>
<td>C</td>
</tr>
<tr>
<td>Minor</td>
<td>D</td>
</tr>
<tr>
<td>No Safety Impact</td>
<td>E</td>
</tr>
</tbody>
</table>

3. Software life cycle data organization

Accordingly to [14], a software life cycle data is any artifact generated during software development. The software life cycle data should support the following actions:
- Describe and record information about a software product during its life cycle;
- Assist usability and maintainability of a software product;
• Define and control life cycle processes;  
• Communicate information about the system, software product or service, and project to those who need it;  
• Provide a history of what happened during development and maintenance to support management and process improvement;  
• Provide evidence that the processes were followed.

A software life cycle data should contain information in the following areas:
• Requirements data: Expected functionalities, operational context, performance constraints and expectations, basis for qualification testing, and key decision rationale.  
• Design data: Architecture, algorithms, design constraints, mapping to requirements, and key decision rationale.  
• Test data: Test strategy and criteria, cases (what to test), procedures (how to carry out tests), test results, and key decision rationale.  
• Configuration data: Configuration description, build instructions, reference to source code, data integrity approach, description of development environment, and key decision rationale.  
• User data: Software overview, system access information, commands and responses, error messages, operational environment, and key decision rationale.  
• Management data: Management plans, status reports, management indicators, criteria and key decision rationale, and contract and other procurement information.  
• Quality data: Quality plans and procedures, corrective action status, root cause analysis, product quality characteristics and process measurement data, and criteria and key decision rationale.

When modifying a software product previously developed per DO-178A using DO-178B, most of the original software life cycle data is impacted and a new baseline of software life cycle data is required. Each software-life cycle data can be classified as:
• (new): life cycle data not requested per DO-178A. This life cycle will cover only the changed portion of the software product. These life cycle data will cover only modified portion of the software product and are requested only per DO-178B.  
• (updated): life cycle data requested per DO-178A and updated to comply with DO-178B. These life cycle data will cover the complete the software product.  
• (previous): life cycle data requested per DO-178A. These life cycle data will cover only unmodified portion of the software product.

The software life cycle data in DO-178B is structured in 8 main areas: Planning, Standards, Requirements, Design, Coding, Verification, Quality Assurance and Certification. This paper provides a methodology to organize the software life cycle data in these areas.

3.1. Planning

DO-178A does not define a Software Development Plan (SDP) some information that is required to be part of a SDP is included in the Support Development System Configuration Document (SDSCD, per DO-178A Section 8.1.9). A new SDP, adherent to DO-178B Section 11.2, is required to show compliance with objectives described in Table A-1 and A-2.

DO-178A does not define a Software Verification Plan (SVP) as a separate document. Some information that is required to be part of a SVP is included in the Software Verification Plan, Procedures and Results (SVPPR, per DO-178A Section 8.1.1). A new SVP, adherent to DO-178B Section 11.3 is required to show compliance with objectives described in Table A-3, A-4, A-5, A-6 and A-7.

Structural coverage is not mentioned in DO-178A. A clear strategy must be presented, based on the fact that structural coverage cannot be partially applied only in the lines of code that were modified. The structural coverage must be applied in the complete software product.

Plan for Software Aspects of Certification (PSAC), Software Configuration Management Plan (SCMP) and Software Quality Assurance Plan (SQAP) must be reviewed to be in compliance with DO-178B Sections 11.1, 11.4 and 11.5. These plans are required to show compliance with objectives described in Table A-1, A-8, A-9 and A-10.

Figure 2 provides a graphical representation on how software plans must be organized to reflect the software life cycle data.

3.2. Standards

DO-178A does not define the software requirements standards (SRS). This document must be defined to address possible requirements that will be created or
changed for the software modification.

A review of the Software Design Standards (SDS) must be performed to assure that all items specified in DO-178B Section 11.7 are addressed.

A new Software Code Standard (SCS), required per DO-178B Section 11.8 is required. The content of the Programmer’s Manual (PM) as required per DO-178A Section 8.1.4, can be included as part of the SCS, in an appendix, for example.

Figure 3 provides a graphical representation on how software standards must be organized to reflect the software life cycle data.

3.2. System and software requirements and design

Software Requirements/Design Process must be performed for the new/changed parts accordingly the SDP.

DO-178B define two levels of requirements (high and low). For the software requirements that will be impacted by a software modification, an assessment must be performed to determine if the impacted software requirements can be classified as high-level requirement or low-level requirement. If is classified as software high-level requirements, so that is impossible to code without a set of additional software low-level requirements. In this case additional software low-level requirements must be generated.

If the impacted software requirement is classified as low-level requirements, an analysis must be performed to determine if this can be treated as derived software low-level requirements or if additional software high-level requirements are needed.

New/changed software high-level requirements must comply with the revised software requirements standards. Additionally, new/changed software low-level requirements must comply with the revised software design standards.

The new software high-level requirements must meet DO-178B Table A-2, objectives 1 and 2 and Table A-3, objectives 2, 3, 4 and 5. The new software low-level requirements must meet DO-178B Table A-2, objectives 4 and 5 and Table A-4, objectives 2, 3, 4 and 5.

Traceability from new/changed software high-level requirements to new/changes system requirements must be achieved, accordingly DO-178B Table A-3 objectives 1 and 6. Another important point is the traceability from new/changed low-level requirements to new/changed software high-level requirements must be achieved, accordingly DO-178B Table A-4 objectives 1 and 6.

The objectives related to software architecture presented on tables A-2, A-3 and A-4 must be assessed for evaluation if additional activities for DO-178B compliance are required.

Figure 4 provides a graphical representation on how system and software requirements and design must be organized to reflect the software life cycle data.
3.3. Software coding process

Software Coding Process must be performed for the new/changed parts according to the SDP. Traceability from new/changed software low-level requirements to new/changed source code must be achieved, accordingly DO-178B Table A-5 objectives 1 and 5.

Source to Object Code Traceability Analysis must be performed considered the new/changed source code items, accordingly DO-178B Section 6.4.4.2.

The objectives related to source/executable object code presented on tables A-5 and A-6 must be assessed for evaluation if additional activities for DO-178B compliance are required for the new/changed source code.

Figure 5 provides a graphical representation on how software code must be organized to reflect the software life cycle data.

![Figure 5. Code life cycle data](image)

3.4. Verification

Software Verification Process must be performed for the new/changed parts according to the SVP.

Requirements Based Tests must be conducted for the new/changed software high-level requirements and software low-level requirements, considering DO-178B, Table A-7, objectives 1, 2, 3 and 4.

Structural Coverage in DO-178A only addresses statement coverage. The new/changed code must be tested to assure that statement coverage is achieved accordingly DO-178B, Table A-7, objective 7.

The MCDC must be performed to the new/changed code to assure that DO-178B, Table A-7, objectives 5 and 6 are achieved.

The data and control coupling analysis must be performed to the new/changed code to assure that DO-178B, Table A-7, objective 8 is achieved.

It is not easy to address the use of structural coverage introduced in DO-178B. It is very difficult to provide evidences of coverage only in the new/changed code. It is recommended to address structural coverage analysis in the complete software product.

Figure 6 provides a graphical representation on how software verification must be organized to reflect the software life cycle data.

![Figure 6. Verification life cycle data](image)

3.5. Quality assurance and certification

Software Quality Assurance Process must be performed for the new/changed parts accordingly the revised SQAP.

Figure 8 provides a graphical representation on how software quality assurance and certification must be organized to reflect the software life cycle data.

![Figure 7. Quality assurance life cycle data](image)

Certification liaison process will be performed by the applicant. In this case the PSAC must expand in this sense. The compliance with DO-178B, Table A-10, objectives 1, 2 and 3 must be addressed in the revised PSAC and communicated and agreed with the certification authority.

The software life cycle data requested by DO-178A is the same as DO-178B. The PSAC, SCI and SAS are the only documents requested to be delivered to certification authorities as part of the aircraft type design data. Although the life cycle data remain unchanged, the content of each document was
expanded and additional information is requested in DO-178B. Figure 9 provides a graphical representation on how software certification must be organized to reflect the software life cycle data.

Figure 8. Software life cycle Data (certification)

3.6. Configuration management

Software Configuration Management Process must be performed accordingly the revised SCMP.

Software life cycle data generated for the previous approved software per DO-178A related to the non-changed parts (software requirements, source code…) must be stored considering Configuration Control Categories defined in DO-178B, Table 7-1.

Software life cycle data generated for the new/changed software affected by the major modification must be stored considering Configuration Control Categories defined in DO-178B, Table 7-1.

As presented in section 3.4, it is not easy to address the use of structural coverage introduced in DO-178B. It is very difficult to provide evidences of coverage only in the modified portion of the software product.

It is encouraged to re-run the complete software product against software requirements (high and low-level) on instrumented code iteratively to acquire the structural coverage data. After that proceed in the following steps:

- Analyze the coverage data and identify the reasons for structural coverage gaps.
- Raise problem reports as necessary to fix the coverage gaps due to insufficiency in the requirements or in the test cases.
- Re-run the tests on instrumented source code to ensure that desired level of structural coverage has been achieved.

These steps will provide structural coverage completeness as required by DO-178B.

3.7. Tool Qualification

DO-178A does not address tool qualification. Accordingly to [13], if processes of DO-178B can be eliminated, reduced or automated by a qualified tool. That means the output from the tool will be used to either meet an objective of DO-178B.

If a tool qualification is applicable, the qualification activities must follow the tool qualification instructions provided in DO-178B and Order 8110.49 Chapter 9.

4. Conclusion

As a general conclusion, the following topics can be considered as high-level activities that summarize the content of the proposed strategy in this paper.

- Evaluate compliance gaps to DO-178B.
- Create a traceability table from DO-178B sections and objectives to software plans and assess possible traceability roles.
- Provide visibility and negotiate approval of PSAC earlier.
- Establish a good communication with certification authorities.
- Plan actions and execute plan.
- Organize Software Life cycle Data, mainly software plans.
- Close gaps and document method and results in the SAS.

As presented in section 3.4, it is not easy to address the use of structural coverage introduced in DO-178B. It is very difficult to provide evidences of coverage only in the modified portion of the software product.
5. References


