Frédéric Pothon
ACG Solutions

DO-330/ED-215 Benefits of the New Tool Qualification Document

January 2013

Contributions and Reviews

Laurent Pomies
DOSoft-Consulting

Cyrille Comar
AdaCore

Hervé Delseny
Airbus

Ben Brosgol
AdaCore

Content

1. Purpose of This Document ____ *3*

2. Need for a Tool Qualification Document ____ *4*

3. Tool Qualification Criteria for Airborne Domain ____ *5*

4. Principles and Technical Aspects ____ *7*

4.1 Domain independence ____ 7

4.2- Identification of Tool Stakeholders ____ 7

4.3- Operational environment is the "target" ____ 8

4.4- Clarification of Requirements for Tools ____ 8

4.5- Need for Tool Validation ____ 9

4.6- A New Table for User Objectives ____ 11

4.7-How to address external components ____ 12

4.8- Robustness aspects ____ 12

5. How to Qualify the Tools? ____ *14*

5.1- Tool user and tool developer processes ____ 14

5.2- TQL-5 versus "Verification" tools ____ 15

5.3- A convenient approach for COTS tools ____ 16

5.4- Improvements for previously qualified tools ____ 17

5.5- Protection and multi-function tools ____ 18

5.6- Use of Service History to qualify a tool ____ 19

5.7- Need for tool qualification in the framework of the Tool life cycle ____ 20

5.8- Use a DO-178C/ED-12C supplement to qualify a tool ____ 21

6. Certification Credit for a Qualified ACG ____ *22*

7. Supporting Information ____ *24*

1. Purpose of This Document

As part of the DO-178C/ED-12C revision effort, a new document *Software Tools Qualification Considerations* (DO-330/ED-215) was developed. Its goal is both to replace the software tool qualification guidance of DO-178B/ED-12B and also to enable and encourage the use of this "mature" guidance outside the airborne domain. Since it may be used independently, DO-330/ED-215 is not considered as a supplement to DO-178C/ED-12C; it is thus titled differently from the specialized technology supplements.

The purpose of this document is to describe how DO-330/ED-215 impacts the current tool qualification approach of DO-178B/ED-12B and how it provides more relevant guidance for both tool users and tool providers.

We first review the rationale for a Tool Qualification document. But before the application of DO-330/ED-215, a fundamental pre-condition is to establish for the project the tool qualification criteria and the Tool Qualification Levels (TQLs). As an example, we show how DO-178C/ED-12C determines the criteria and TQLs for the airborne domain. In this domain, the criteria are based on the possible impact of a tool error on the software life cycle processes.

We then highlight the main impact of DO-330/ED-215 on current practice, and provide the relevant information to help the reader to apply this new guidance.

Some supporting information is provided in an appendix of DO-330/ED-215. We describe one of the most important topics, addressing the possible certification credit when using a qualified AutoCode Generator (ACG).

2. Need for a Tool Qualification Document

Since automated tools are (potentially) more reliable than humans at performing certain types of analysis, SC-205/WG-71 wanted to encourage their usage, provided that appropriate assurance could be obtained that the tools are at least as dependable as the manual processes that they are replacing. This approach necessitated developing clear guidance for qualifying the software tools. But there was no reason to restrict such considerations to the “airborne domain”. A tool vendor might apply a single qualification process to a tool that could be used in multiple domains, resulting in a wider selection of tools with increased tool quality.

For these reasons, the concept of a DO-178C/ED-12C “supplement” would not be appropriate for tool qualification guidance. Instead, tool qualification considerations are the subject of a new domain-independent document: DO-330/ED-215. This document is to be used in conjunction with a domain-specific standard that governs the acceptability of the actual product. To make DO-330/ED-215 applicable, the relevant domain-related document should:

- Identify that DO-330/ED-215 is applicable
- Define its own tool qualification criteria
- Define, based on these criteria and other considerations if necessary (e.g. the reliability of the product) the selection of the tool qualification level (TQL-1 to TQL-5)

For airborne software, the new tool qualification criteria are described below. Each domain is free to define its own tool qualification criteria.

Then, once the domain has defined the applicable criteria, DO-330/ED-215 applies. Therefore, objectives to be satisfied for each TQL are defined, independently of the domain, and of the qualification criteria.

At first glance, DO-330/ED-215 looks like DO-178/ED-12 itself. This is because DO-178/ED-12 was used as the basis of the development of this new document. But the text was adapted to be directly applicable to tools, and to address all the tool aspects.

3. Tool Qualification Criteria for Airborne Domain

Section 12.2.2 in DO-178C/ED-12C defines three criteria that determine the applicable tool qualification level (TQL) with regard of the software level.

> a. Criteria 1: A tool whose output is part of the airborne software and thus could insert an error.
>
> b. Criteria 2: 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 that automated by the tool, or
>
> 2. Development process(es) that could have an impact on the airborne software.
>
> c. Criteria 3: A tool that, within the scope of its intended use, could fail to detect an error.

Criterion 1 subsumes what were called “development tools” in DO-178B/ED-12B, while the two other criteria split the former “verification tools” depending on the certification credit claimed by the qualification of the tool.

Criterion 3 is the “classic” use of a verification tool: The purpose of the tool is to produce or verify an artifact, and the certification credit claim is only on objectives applicable to this artifact.

Examples:

- A tool that produces test procedures from test cases; the certification credit is limited to DO-178C/ED-12C objective A7-1 (“Test procedures are correct”).
- A code checker that verifies the compliance of source code to the coding standard; the certification credit is limited to DO-178C/ED-12C objective A5-4 (“Source code conforms to standards”).

For Criterion 2, the certification credit claimed is extended to objectives that are beyond the data directly verified by the tool.

In an appendix of DO-330/ED-215, FAQ D.5 provides additional rationale for these 3 criteria and also some examples of the difference between Criteria 2 and 3, using a “proof tool” and a “static code analyzer”.

> a. Example 1: A proof tool may be used to automate some verification of Source Code. Criteria 3 could be applied based on this tool’s use and credit claimed. However, if the applicant claims that testing activity to detect a class of error becomes unnecessary based on the tool detecting the related class of error, then the criteria 2 becomes applicable. In this case, it corresponds to “a reduction of software verification process(es) other than that automated by the tool.”
>
> b. Example 2: A static code analyzer may be used to automate some verification of Source Code review. Criteria 3 could be applied based on this tool’s use and credit claimed. However, if the applicant claims to not include some specific mechanisms in the resulting software in order to detect and treat the possible overflow, and run-time errors based on the confidence on the tool, then the criteria 2 becomes applicable. In this case, it corresponds to “a reduction of software development process(es).”

One of the main principles of DO-178C/ED-12C is to require multiple verification filters to improve the error detection. The certification credit claimed with the application of criteria 3 is equivalent to remove one filter. This filter is considered as useless (in term of error detection) as the verification performed by the tool is claimed as thorough enough to detect the possible errors. That's why for these tools the Tool Qualification Level (TQL) is higher than for a "classic" verification tool,

The applicable TQL is defined in DO-178C/ED-12C Table 12-1, based on the qualification criterion and the software level:

Software Level	Criteria		
	1	2	3
A	TQL-1	TQL-4	TQL-5
B	TQL-2	TQL-4	TQL-5
C	TQL-3	TQL-5	TQL-5
D	TQL-4	TQL-5	TQL-5

The TQL applicable for Criterion 1 is the replacement of DO-178B/ED-12B development tool for each software level, while TQL-5 for Criterion 3 is the replacement of DO-178B/ED-12B verification tool.

The TQL applicable for Criterion 2 basically requires ahigher level of rigor for tools used on software at level A or B, in order to increase the confidence in the use of the tool (that is, TQL-4 instead of TQL-5). TQL-4 requires that the Tool Requirements data describe all functionality implemented in the tool and provide additional detail about the tool architecture. TQL-4 also requires verification that the tool complies with Tool Requirements. TQL-4 objectives are considered as a minimum to claim confidence in the use of such tools. But the purpose of applying TQL-4 for software level A or B (AL1 and AL2 for DO-278A users) is not to prevent the use of this kind of tool. The following approaches may be considered for tool use:

- In case of deficiencies in the tool life cycle data needed to qualify the tool at TQL-4, the applicant may still use the tool and qualify it at TQL-5. Certification/approval credit is limited to the verification objectives of the data under verification.

- In case of COTS, if the data life cycle is not provided by the tool supplier to qualify the tool at level TQL-4, section 11 of DO-330/ED-215 allows an applicant to augment the data in order to satisfy the objectives for the applicable TQL.

An appendix in DO-330/ED-215 provides some additional rationale for no longer using the terms "development tool" and "'verification tool" (FAQ D1).

4. Principles and Technical Aspects

The following sections explain the main principles of DO-330/ED-215.

4.1 Domain independence

A goal of DO-330/ED-215 is to be usable across a variety of application domains. However, since a tool may be qualified only in the scope of a "user context" it was difficult to both find terminology and identify "domain data" that would be relevant for all domains.

The decision was to produce the document for the airborne software domain, which will be the first and probably the main user, and to include a section (§1.3) describing how to apply this document more generally. Section 1.3 explains the need for all domains to define their own tool qualification criteria and tool qualification levels, and to adapt the terminology as appropriate:

> b. Throughout this document terms such as "software life cycle", "software processes", "software plans", and "software" are used to refer to the product life cycle, processes, plans, and domain where the tool will be used (that is, a software domain is used instead of a generic domain). For other domains the word "software" may be replaced by the appropriate domain, such as, "electronic hardware", "system", "aeronautical database", "aviation software", etc.

Appendix B of DO-330/ED-215 illustrates the definition of tool qualification criteria and tool qualification levels. This is just a copy of the section 12.2 of DO-278A/ED-109A (CNS/ATM software). The purpose is to help users from other domains to develop their own tool qualification section.

4.2- Identification of Tool Stakeholders

The purpose of DO-330/ED-215 is to identify all objectives that should be satisfied to qualify a tool in a specific context. It was therefore important to realize that at least two stakeholders are involved in the tool qualification processes: the *tool user*, that is, the team that uses the tool in the software life cycle; and the *tool developer*, who performed all activities to deliver the tool product to the tool user.

Unfortunately, except for the COTS section (§11.3) where the responsibility separation between tool user and tool developer is explicitly defined, the direct use of the actors in the process description is not used. Instead, the responsibility separation is identified in various ways in the document:

- Section §3.2, which provides a description of typical stakeholders

> Tool qualification typically involves multiple stakeholders. In most projects there will be both a tool user and a tool developer. The tool user typically identifies the tool to be used, assesses its impact on the software processes, addresses the use of the tool in the scope of the software process in which the tool is used, and performs the tool qualification within the context of the software approval. The tool developer typically describes the processes of the tool development, verification, and integral processes, and addresses the development of the tool in compliance with the user needs expressed in the Tool Operational Requirements (TOR).

- The terminology used: The term "**operational**" (e.g Tool **Operational** Requirements, Tool **Operational** Verification and Validation process"), is used to identify the "user" perspective.
- Table T-0, which was provided to identify all objectives "*typically*" applicable to the user

4.3- Operational environment is the "target"

The "target" for a software tool could be considered as the environment where the tool will operate in the software life cycle context. This context is referred to as the "Operational Environment" in DO-330/ED-215.

> Tool Operational Requirements – The Tool Operational Requirements can be viewed as the "system requirements" for a tool. Tool Operational Requirements are from the perspective of the user and may not provide all requirements necessary to develop a tool.

DO-330/ED-215 also identifies other environments, used in the context of the tool developer processes:

- The tool development environment, that is, the environment where the tool is developed
- The tool verification environment(s), where the tool in its executable format is verified (tested). This definition includes a strong recommendation that the tool verification environment(s) should be representative of the Tool Operational Environment(s).

As a consequence of these definitions, there is no notion of a "target" environment; however, specific objectives were developed that relate to:

- Installing the tool in the appropriate environment (objective T0-3 for the operational environment and T2-8 for the verification environment);
- Verifying the compatibility of the tool requirements with the operational environment (T3-3);
- Verifying the tool with respect to its operational requirements in the operational environment. (T0-5), and
- Performing validation activities also in the operational environment as described in the next subsection.

4.4- Clarification of Requirements for Tools

In DO-178B/ED-12B, there were some ambiguities in the Tool Operational Requirements definition. Its content is considered as "equivalent to the software requirements":

§12.2.3.c (2) states "*Tool Operational Requirements satisfies the same objectives as the Software Requirements Data*". But they are also used as "system requirements" for the tool: §12.2.1.d "… *since the tool's high level requirements correspond to its Tool Operational Requirements instead of system requirements*".

DO-330/ED-215 clarifies the different tiers of requirements, starting with the "Tool Operational requirements" that described the software life cycle needs. TOR content description is the purpose of the section 10.3.1:

> The Tool Operational Requirements define the tool's functionality and interface from a software life cycle process perspective (that is, the process which uses the tool). The Tool Operational Requirements should include, as applicable:
>
> a. Description of the context of the tool use, including interfaces with other tools and integration of the tool output files into the resultant software.
>
> b. Description of the tool operational environment(s) (where the tool will be installed).
>
> c. Description of input files, including format, language definition, etc.
>
> d. Description of output files, including format and contents.
>
> e. Requirements for all the tool functions and technical features used to satisfy the identified software life cycle process(es).
>
> f. Requirements to address the abnormal activation modes or inconsistency inputs that should be detected by the tool. These requirements should consider the impact of those modes on the functionality and outputs of the tool. (This item is not applicable to TQL-5.)
>
> g. The applicable user information, such as a user manual and installation guide or a reference to it, if not provided as part of the Tool Requirements data.
>
> h. Description of the operational use of the tool (including selected options, parameters values, command line, etc.).
>
> i. Performance requirements specifying the behavior of the tool output.

These Tool Operational Requirements are refined during the Tool Development Processes into one or several levels of "Tool requirements", poorly identified as "Tool requirements" and "tool low-level requirements". Each refinement level may include some derived requirements. Derived requirements are those that are not traceable to the higher level (this is a simpler definition than in DO-178C/ED-12C). They will be evaluated to ensure that they do not impact the expected functionality and outputs defined in the Tool Operational Requirements.

The TOR might not document all tool functions; it only needs to treat those required by the user. This is not the case for the Tool Requirements, which need to describe all tool functions and features. Any extraneous functions will then be identified as derived requirements, and then analyzed.

4.5- Need for Tool Validation

Software Requirements validation is out of the scope of DO-178C/ED-12C, it is under the responsibility of the system processes. But in the context of DO-330/ED-215, it is necessary to assess that the tool is compliant with user requirements, whether or not explicitly defined in the TOR. This is the purpose of "validation".

So in addition to the verification objectives, validation is the purpose of two complementary objectives:

- Objective T0-6: validate the Tool Operational Requirements by review and/or analyses. The goal of this activity is to check the completeness and relevance of the requirements with respect to the certification credit claimed.
- Objective T0-7: validate the behavior of the tool in the operational environment, by execution (tests), in order to assess that all needs of the software life cycle are met.

§6.2.1 states:

> The validation objectives of the tool operational verification and validation process consist of the analysis of the functionality and the outputs of the tools for correctness and completeness with respect to the software life cycle activities performed. Validation objectives are:
>
> aa. Ensure that the Tool Operational Requirements are sufficient and correct to eliminate, reduce, or automate the process(es) identified in the PSAC.
>
> bb. Ensure that the tool meets the needs of the software life cycle process in the tool operational environment.

These two objectives supplement the verification objectives performed on the Tool Operational Requirements and on the Tool itself for compliance with the Tool Operational Requirements. That's why DO-330/ED-215 §6.2 identifies the objectives and activities of the "Tool Operational Verification and Validation Process".

4.6- A New Table for User Objectives

There are ten objective tables in DO-178C/ED-12C, but eleven in DO-330/ED-215! Here is the additional table:

	Objective		Activity	Applicability by TQL					Output		Control Category by TQL				
	Description	Ref.	Ref.	1	2	3	4	5	Description	Ref.	1	2	3	4	5
	Planning Process														
1	The tool qualification need is established.	4.1	[Note 1]	❍	❍	❍	❍	❍	Tool-specific information in the Plan for Software Aspects of Certification	10.1.1	①	①	①	①	①
	Tool Operational Requirements Process														
2	Tool Operational Requirements are defined.	5.1.1.a	5.1.2.a 5.1.2.b 5.1.2.c	❍	❍	❍	❍	❍	Tool Operational Requirements	10.3.1	①	①	①	①	②
	Tool Operational Integration Process														
3	Tool Executable Object Code is installed in the tool operational environment.	5.3.1.a	5.3.2.a 5.3.2.b 5.3.2.c	❍	❍	❍	❍	❍	Tool Executable Object Code Tool Installation Report	10.2.4 10.3.2	② ②	② ②	② ②	② ②	② ②
	Tool Operational Verification and Validation Process														
4	Tool Operational Requirements are complete, accurate, verifiable, and consistent.	6.2.1.a	6.2.2.a	●	●	❍	❍		Tool Operational Verification and Validation Results	10.3.4	②	②	②	②	
5	Tool operation complies with the Tool Operational Requirements.	6.2.1.b	6.2.2.c	●	●	❍	❍	❍	Tool Operational Verification and Validation Cases and Procedures Tool Operational Verification and Validation Results	10.3.3 10.3.4	② ②	② ②	② ②	② ②	② ②
6	Tool Operational Requirements are sufficient and correct.	6.2.1.aa	6.2.2.b	●	●	❍	❍	❍	Tool Operational Verification and Validation Results	10.3.4	②	②	②	②	②
7	Software life cycle process needs are met by the tool.	6.2.1.bb	6.2.2.c	❍	❍	❍	❍	❍	Tool Operational Verification and Validation Cases and Procedures Tool Operational Verification and Validation Results	10.3.3 10.3.4	② ②	② ②	② ②	② ②	② ②

This table was created to identify all objectives addressing the use of the tool in the software life cycle processes. This table (for “Tool Operational Processes) thus identifies objectives for:

- Planning process: To define the need for qualification and the applicable tool qualification level. Typically this information is provided in the PSAC
- Development process: To develop the Tool Operation Requirements
- Integration process: To install the tool in the Tool Operational Environment
- And the four objectives of the Tool Operation Verification and Validation process.

4.7-How to address external components

Application of DO-178B/ED-12B to development tools for software level A raised concerns on object code to source code traceability and the need for additional verification on object code. There is no further consideration about the object code in DO-330/ED-215. But additional considerations on “external components” were added.

This term is defined in the glossary:

> External components – Components of the tool software that are outside the control of the developer of the tool. Examples include primitive functions provided by the operating system or compiler run-time library, or functions provided by a COTS or open source software library.

Examples are also provided in the FAQ C.2 in an appendix of DO-330/ED-215.

To address these external components, several new objectives are defined:

- In the design process (§5.2.2.g): The description of the interface should identify all the external components, such as file management routines, primitives, memory allocation calls, and routines supporting the user interface management (for example, command line or display message).
- The correctness of their identification and of their interfaces are verified during the Tool Architecture review and analyses (§6.1.3.3.e). This is applicable for TQL-1 and 2.
- The requirements-based test coverage analysis should also verify that the requirements based tests exercise the interface and the functionality of each function of the external components utilized by the tool. This is applicable only for TQL-1.

4.8- Robustness aspects

The robustness aspects for a tool were clarified. The robustness test cases should be requirement based. For that purpose, the Tool Requirements should identify the failure modes and define the tool responses. The goal was to prevent the generation of wrong outputs.

The verification of the tool requirements includes the completeness and consistency of the requirements to address the failure modes.

Objectives T6-2 and T6-4 (“Tool Executable Object Code is robust with Tool Requirements/ with low-level tool requirements”) are satisfied by developing test cases from the Tool Requirements (and low-level requirements if any) identifying the failure modes

In addition, it was also agreed that a general behavior may be defined, without identifying specific failure modes. In such a case, some additional test cases should be developed to complete the demonstration that the tool can properly deal with abnormal conditions or data. Here is the corresponding text (§6.1.4.2) concerning Tool Testing Activities for robustness aspects:

c. Robustness tests should be performed to address all failure modes (for example, abnormal activation modes, inconsistency inputs, etc.) identified in Tool Requirements.

d. If necessary, additional robustness tests should also be developed to complete the demonstration of the following:

- The ability of the tool to respond to abnormal inputs or conditions.
- The detection of abnormal behavior.
- The prevention of invalid output.

5. How to Qualify the Tools?

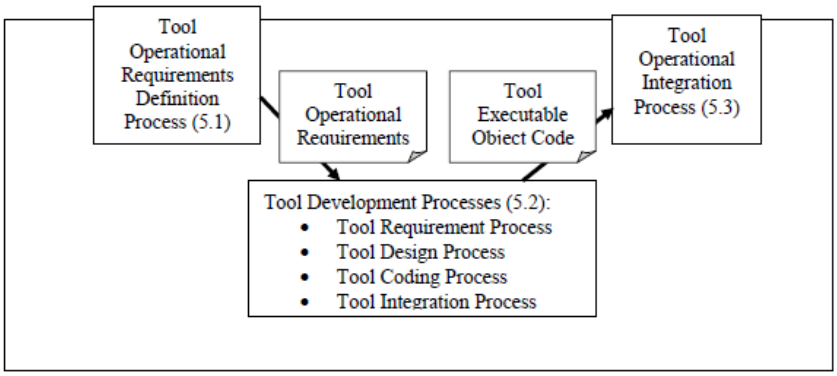
5.1- Tool user and tool developer processes

Complementary processes are defined for the tool user and the tool developer:

- Planning process
    - **Tool User**: The user should identify the need and level of qualification for the tool and provide rationale for the certification credit claimed. DO-330/ED-215 thus identifies the specific information to be provided in the PSAC, consistent with DO-178C/ED-12C. Although this is well known in airborne domain, the domain-independent nature of DO-330/ED-215 made it imperative to emphasize that a similar approach is necessary for all domains. In addition the PSAC should also describe (or reference) the tool-related activities to be performed by the user.
    - **Tool Developer**: Except for the description of user activities that are the purpose of data provided by the Tool User, the Tool Developer provides plans and standards to satisfy all objectives of the planning process.

- Development process
    - **Tool User**: A tool is developed to address the needs of the software life cycle to automate one or several tasks. These needs should be defined, typically by the user, in the Tool Operational Requirements. This is the "*Tool Operational Requirements Definition Process*.
    - **Tool Developer**: The tool is developed from the Tool Operational Requirements in compliance with the Tool Life cycle defined in the Tool Development Plan. This is typically based on the specification, design, coding, and integration processes. The "integration process" is here limited to the production of the Tool Executable Code.
    - **Tool User**: After delivery of a release of the tool, the user installs the tool in the "Tool Operational Environment": This is the "Tool Operational Integration Process".

The figure 5-1summarizes these complementary development process



- Verification (and validation) process

    - **Tool Developer**: All verification objectives to be satisfied are similar to those specified in DO-178C/ED-12C: verification of output of the planning process, specification, design, coding, and integration process. Then tests are performed in the tool verification environment, followed by test data verification including requirement and structural coverage analysis.
    - **Tool User**: In addition, the Tool Verification and Validation process activities are performed in the Tool Operational Environment. As a consequence, the compliance of the tool with its operational environment is addressed by tool user activities. This approach will normally facilitate the qualification renewal when the Tool Operational Environment changes (e.g. upgrade of workstation).

- SQA and SCM processes: DO-330/ED-215 does not separate the objectives for these processes between tool user and tool developer. However, as the planning, development, and verification process are composed of complementary activities, to satisfy the SCM and SQA objectives an organization should be set up to manage and oversee the complete life cycle processes, in the context of both the tool user and the tool developer.

- Qualification liaison process: The objectives of the Tool Qualification Liaison process are based on the complementary data provided by both the tool developer and the tool user. The data provided should address the complete life cycle processes, regardless of the packaging. As the qualification is claimed for each system, the Tool User is responsible of this process.

5.2- TQL-5 versus "Verification" tools

As defined in chapter 3, TQL-5 is equivalent to the qualification level for "verification tools" in DO-178B/ED-12B. The initial intent was to keep the same level of rigor for these tools, to not prevent their use (i.e., not "raise the bar"). Thus the objectives applicable to TQL-5 should be equivalent to the qualification criteria of DO-178B/ED-12B for a verification tool: "*the tool complies with its Tool Operational Requirements under normal operational conditions*" (§12.2.2). They also have to include the applicable objectives of the other integral processes, as defined in §12.2.c " "*the software configuration management process and software quality assurance process objectives should apply*"

DO-330/ED-215 provides more accurate and complete guidance for tools at TQL-5 than DO-178B/ED-12B did for verification tools. The intent is not to ask for more activities or more data. The qualification does not require any data from the tool development process. That means that it should be still possible to qualify a tool at TQL-5 without any data from the Tool Developer (e.g., the tool vendor for a COTS tool). However, it clarifies the content of the TOR, the compliance of the tool with the resulting software process needs, and the objectives of other integral processes applicable to TQL-5.

The objectives associated with TQL-5 are mainly in Table T-0. This clarifies that it is still possible to qualify a tool at TQL-5 without any data from a tool vendor. All objectives are "user oriented".

The content of the Tool Operational Requirements is clarified. And beyond this clarification, the validation objectives create a relationship between the TOR and the certification credit claimed. Evidence should be provided that the tool, installed in the Tool Operational Environment, satisfies all of the needs of the software process.

Additionally, some objectives of the other integral process (SCM, SQA and qualification liaison) are applicable at TQL-5: identification of configuration items and archive for the SCM process. Assurance is obtained that the tool processes comply with approved plans and conformity review for SQA. Note that this conformity review may be part of a software process.

However, there is a new objective in the table on the Qualification Liaison process, applicable at all levels: to analyze the known problems for possible impact on the Tool Operational Requirements. This may appear somewhat difficult in the absence of data from the tool vendor. But the committee felt that this analysis should be conducted to identify possible tool limitations that might reduce the certification credit claimed.

Details on “Verification Tool” Qualification Improvements” appear in a FAQ in an appendix of DO-330/ED-215 (FAQ D.6).

5.3- A convenient approach for COTS tools

One of the goals of DO-330/ED-215 was to facilitate and to clarify the qualification of commercial tools.

The approach for qualifying COTS tools exploits the definition of stakeholders and the definition of the complementary processes between tool user and tool developer. But the main problem when trying to apply the tool qualification guidance is that a COTS tool is **not** developed from Tool Operational Requirements defined by a user.

Section §11.3 of DO-330/ED-215 therefore describes a possible means for satisfying the tool qualification objectives in the case of a COTS tool. Section §11.3 addresses this issue by separating the TOR content into two parts:

- A developer-TOR that is used in developing the tool. It is also used for all verification activities, e.g. verifying the compliance and traceability of Tool requirements.
- The developer-TOR is supplemented by the TOR provided by the Tool User. The TOR includes or references the developer-TOR, and provides additional information and limitations for the software life cycle processes. This additional information are used for the validation activities

Similarly the Tool Developer also provides a developer-TQP, developer-TCI and developer-TAS limited to its activities, and the Tool-User provides the TQP, TCI and TAS.

Based on this separation, section §11.3 provides tables for typical objectives to be satisfied by the Tool User, and those to be satisfied by the Tool Developer. It also provides typical content of the data shared between the two stakeholders.

Here is an overview of this separation:

Table T-0 TOOL OPERATIONAL PROCESSES			
1	The tool qualification need is established.	TOOL USER	
2	Tool Operational Requirements are defined.	SHARED:	*Developer develops the developer-TOR* *User -TOR supplements the developer-TOR to produce the TOR*
3	Tool Executable Object Code is installed in the tool operational environment	TOOL USER	
4 and 5	Tool Operational Verification objectives	TOOL DEVELOPER	*Based on the developer-TOR*
6 and 7	Tool Operational Validation objectives	TOOL USER	*Based on the TOR*
T-1 : TOOL PLANNING PROCESS			
1	Tool development and integral processes are defined.	SHARED	*Application limited to the scope of each stakeholder*
2	Transition criteria, inter-relationships, and sequencing among processes of tool processes are defined.	SHARED	*Application limited to the scope of each stakeholder*
3	Tool development environment is selected and defined.	TOOL DEVELOPER	
4	Additional considerations are addressed	SHARED	*Application limited to the scope of each stakeholder*
5	Tool development standards are defined.	TOOL DEVELOPER	
6 and 7	Plan review objectives	SHARED	*Application limited to the scope of each stakeholder*
T-2 : TOOL DEVELOPMENT PROCESS			
All	TOOL DEVELOPER		
T-3 to T-7: TOOL VERIFICATION PROCESS			
All	TOOL DEVELOPER		
T8 and T-9 : SCM and SQA PROCESS			
All	SHARED	Application limited to the scope of each stakeholder	
T-10 QUALIFICATION LIAISON PROCESS			
All	TOOL USER		

5.4- Improvements for previously qualified tools

In the “additional considerations” section of DO-330/ED-215 the reuse issue is addressed in “Previously Qualified Tools” (§11.2).

Guidance is provided for three aspects of tool qualification:

1- Reuse of previously qualified tools that are unchanged

In this paragraph, the document provides criteria to be analyzed to be sure that the tool is suitable for reuse without any change. The criteria include the applicable TQL (same or lower); no change in the data, operational requirements and environments; same version.

2- Changes to the tool operational environment

This paragraph is probably the most important as it explains that in case of a change in the operational environment only (e.g. upgrade of the workstation), the impact analysis may be limited to a demonstration that the tool verification environment is representative of the tool operational environment, and to an analysis of the tool operational verification and validation processes. Therefore, such changes may be assessed by user activities only, independently of the tool developer.

3- Changes to the tool itself

In such cases, the impact analysis should identify any needed re-verification activities.

5.5- Protection and multi-function tools

Initially the question was "what partitioning" means for tools?" This concept was considered to be not directly applicable to tools, but it may sometimes be necessary to guarantee a form of isolation between tools or between tool functions to prevent the presence of common errors.

After an extended discussion, the term "protection" was used and defined in the glossary:

> Protection – The use of a mechanism to ensure that a tool function cannot adversely impact another tool function.

The concept of protection is applicable when different levels of qualification are proposed for different tool functions. This is assessed during the planning process when determining the need for tool qualification (§12.2.1 of DO-178C/ED-12C).

> The tool qualification process may be applied to a single tool, a collection of tools, or one or more functions within a tool. For a tool with multiple functions, if protection of tool functions can be demonstrated, only those functions that are used to eliminate, reduce or automate software life cycle processes, and whose outputs are not verified, need be qualified. Protection is the use of a mechanism to ensure that a tool function cannot adversely impact another tool function.

Further details are supplied in an appendix to DO-330/ED-215 in a FAQ (FAQ C.1 "*What Does "Protection" Mean for Tools and What Are Some Means to Achieve It?"*

This FAQ extends the concept of protection to apply to two tools (not only to tool functions), and it lists (without pretending to be exhaustive) some possible techniques:

- spatial and temporal partitioning
- functional partitioning
- functional deactivation

If applicable, the protection mechanism needs to be documented:

- Tool planning process: The methods used to verify the integrity of protection need to be provided in the Tool Verification Plan. These methods may be a combination of review, analyses, and tests.
- Tool development (design) process: The tool architecture describes the protection mechanism

- Tool Verification process: The specific objective of tool architecture verification (§6.1.3.3.d and T-4 objective 10) applicable from TQL-1 to TQL-4 needs to be satisfied: "*Protection mechanisms, if used, are confirmed*"

Based on this concept, there are two possible applications:

1- **Multi-function tools** are described in section 11.1 of DO-330/ED-215. This section is applicable when an applicant proposes to qualify only some functions of the tool, or not all functions at the same level. This approach is possible only if a protection mechanism is used. In this case, the purpose of the protection is to ensure that the outputs of the functions (or groups of functions) qualified at a lower TQL have no effect on the output of the other functions.

But an important note is added that the guidance of this section, including protection, is applicable only when the tool contains functionality above TQl-5.

2- **Verification of the outputs of a non-qualified tool**: Two separate tools, or a multi-function tool, might both produce an output and verify this same output. In such cases, the goal of the protection mechanism is to avoid an error that might affect both functions. Note that when the verification objectives satisfied by the use of the tool are required to be demonstrated with independence, a higher degree of protection (called "independence between tools) will be required.

When a qualified tool is used to verify the outputs of an unqualified tool, the discussion in FAQ D-7 applies. The main concern was about the ability of qualified tools to satisfy all objectives applicable to the outputs of unqualified tools.

This FAQ discusses the following considerations:

- Coverage of verification objectives that apply to the unqualified tool's output
- Operating conditions of the qualified tool, such as configuration and setup
- Common cause avoidance (that is, avoiding a single error affecting both the unqualified tool and the qualified tool). This could be satisfied through separate teams, separate tool development processes and/or dissimilar technical approaches. The FAQ also addresses the problem of using common components (such as libraries) in both tools
- Protection between the tools (that is, avoiding interference of the unqualified tool on the qualified tool's proper operation).

5.6- Use of Service History to qualify a tool

The guidance of DO-330/ED-215 about the use of service history to qualify a tool is equivalent to the one in DO-178C/ED-12C for software. But despite the clarifications in DO-178C/ED-12C, it might be still very difficult to claim credit with this "alternate means" for software.

For a tool, the situation could be different.

Section 11.4 explains that one possible reason to use service history is to increase the TQL. This could arise for a Criterion 2 tool that requires the application of TQL-4 for software level A and B instead of TQL-5 for level C and D, or for a Criterion 3 tool. But the difference between the two criteria is only based on the certification credit claimed through the qualification of the tool, not on the tool functions.

An acceptable approach may thus be to first qualify a tool applying Criterion 3 (with the limited certification credit), so at TQL-5. During operational use, the tool service history may

be recorded. This service history may reveal that some software verification activities never detected any errors. Based on this evidence the Tool User may propose increasing the certification credit of the tool, to eliminate unnecessary verification activities. This corresponds to qualifying the tool at TQL-4 based on the application of Criterion 2 (for software level A or B).

This qualification at level TQL-4 will be based on qualification at TQL-5 supplemented by additional data from the tool's service history.

5.7- Need for tool qualification in the framework of the Tool life cycle

It may seem strange to need to qualify a tool for qualifying a tool. But if you look more closely, it is good practice to automate activities that satisfy the objectives for qualifying a tool, especially for TQL-1 to 3. To use the DO-178B/ED-12B wording, you may qualify some "verification tools" to automate/reduce/alleviate the processes of qualifying a "development tool".

In DO-178B/ED-12B the approach is required due to the "recursiveness" of the tool guidance: to qualify a development tool, the same objectives as the software should be satisfied … including additional considerations, and thus the tool qualification section.

In DO-178C/ED-12C, the tool qualification section first identifies the need for qualification and the applicable TQL, and then refers to DO-330/ED-215. Specific guidance is provided in DO-330/ED-215 to address the need for tool qualification of a tool used in the Tool life cycle.

The tool planning process objective concerning the need to address additional considerations (T1-4) explicitly includes the assessment of "the need to qualify any tool(s) used in the framework of the tool life cycle processes" (4.3.d).

The Tool Planning process activity details this assessment and the applicable TQL. In this context the only two criteria that are kept are equivalent to those of DO-178B/ED-12B:

- A tool that may inject an error is to be qualified at the same level as the tool to be qualified itself
- A tool that may only fail to detect an error is to be qualified at TQL-5

This is defined in §4.4.e

> e. An assessment on all the tools used in the framework of the tool life cycle processes should be conducted in order to identify the need for qualification of these tools. Qualification of these tools is needed when processes of this document are eliminated, reduced, or automated by the use of a tool without its output verified as specified in section 6. For a tool that can introduce an error in the outputs of a tool, the applicable TQL is the same as the tool being developed. For a tool that cannot introduce an error in the output of the tool, but may fail to detect an error in the tool life cycle data, the applicable TQL is TQL-5.

Note that Criterion 2 is not applicable to the "second layer" tools!

5.8- Use a DO-178C/ED-12C supplement to qualify a tool

The various DO-178C/ED-12C supplements explain that one supplement may be used in conjunction with any other. But there is no similar text in DO-330/ED-215! If the technology addressed in one or more of the supplements is used in the development of the tool, may the corresponding supplement(s) be used to help qualify the tool?

DO-330/ED-215 is domain independent. To make this document applicable, a domain-dependent document, such as DO-178C/ED-12C, should reference it. But this independence would be violated if DO-330/ED-215 were to reference the DO-178C/ED-12C supplements.

Are these supplements applicable to other domains? We don't know at this point. But to qualify a tool in the airborne or CNS/ATM software domains, the supplements should be applicable.

However the supplements add, delete or otherwise modify guidance (objectives, activities, and software life cycle data) of DO-178C/ED-12C; the impact on the guidance of the tool qualification may be not always direct.

To use a supplement for qualifying a tool, it is thus necessary to perform several activities during the Tool Qualification planning process:

- Review all potentially relevant supplements and identify those that will be used;,
- Identify the impact of the use of the selected supplement on the tool qualification objectives to be satisfied;
- Document in the TQP the means to satisfy all the objectives, as adapted by the supplement where applicable.

6. Certification Credit for a Qualified ACG

The principal and perhaps only example of a Criterion 1 tool (or a DO-178B/ED-12B Development tool) is an AutoCode Generator (or an analogous tool that generates configuration files or parameter data). The qualification of such tools requires a significant effort, similar to the software itself. But it is not clear what certification credit may be claimed for such a tool.

After a difficult and extended discussion, a Discussion Paper was approved on this topic! It appears in DO-330/ED-215 as FAQ D.8 in Appendix D.

The purpose of this FAQ is to clarify under which conditions some certification/approval credit (satisfaction of objectives) may be claimed when using a qualified ACG. Based on some typical scenarios, it provides insight into the thought processes and potential considerations to be addressed when using a qualified ACG.

- ***FAQ D.8: How Might One Use a Qualified Autocode Generator?***

  First it was important to clarify what an ACG does not do: a code generator should not be used to verify a model: The purpose of a code generator is to translate a model into source code, and the purpose of the qualification is to provide confidence in the completeness and correctness of this translation ("*What is in the model is in the code*")

  The "correctness and completeness" of the translation is mainly based on having an accurate set of tool requirements and a sound tool verification process:

  - All the translation rules from input files to the Source Code are defined in the Tool Requirements.
  - The correctness of the translation rules is verified through Tool Requirements verification,
  - The correctness of their implementation is verified through the tool testing process and the tool operational verification and validation process.

  Therefore, provided that the Tool Requirements are accurate and the tool verification activities are complete and relevant to the Source Code verification objectives, credit may be claimed for DO-178C/ED-12C Table A-5, objectives 1 to 6, with the following limitations:

  - Part of objective 6 of Table A-5, worst-case execution time or stack usage analyses, may only be satisfied after the Source Code generation.
  - It is also necessary to verify that the tool has been exercised on the complete set of input files to ensure that all the low-level requirements have been developed into Source Code (Table A-5, objective 5).

  The FAQ proposes three scenarios of qualified ACGs to satisfy Table A-6 objectives 3 and 4 (that is, Executable Object Code complies with (normally and robustly) the low-level requirements) and Table A-7 objectives 1, 2 and 4.

  - Scenario 1 - Satisfaction of low-level requirements-based test objectives through test cases based on the low-level requirements.
  - Scenario 2 - Satisfaction of low-level requirements-based test objectives through test cases based on the requirements from which the model (input files) is developed.
  - Scenario 3 - Satisfaction of low-level requirements-based tests objectives through qualification of the ACG and verification of a set of representative input files.

For each scenario the FAQ identifies whether the objectives are satisfied through tool qualification processes ("tools"), through airborne software processes ("Software"), or through the coverage involved in the two processes ("tool/software").

Objectives	ED-12C/ ED-109A Table A-5 (Obj 1-4)	ED-12C/ ED-109A Table A-5 (Obj 5, 6)	ED-12C /ED-109A Table A-6 (Obj 3, 4)	ED-12C /ED-109A Table A-7 (Obj 1, 2, 4)	ED-215 Table T-0 (Obj 5, 7)	ED-12C /ED-109A Table A-7 (Obj 5-7)
Scenario 1	Tool	Tool/Software	Software	Software	Software (Note 2)	Software
Scenario 2	Tool	Tool/Software	Software	Software	Software (Note 2)	Tool/Software
Scenario 3	Tool	Tool/Software	Tool	Tool/Software	Tool	Tool/Software

The main idea is to consider that either the activities are performed for every software version (recurring activities), or only once during the tool qualification process (non-recurring activities). When the objectives are satisfied through the tool qualification processes, it is assumed that the activities performed in the scope of the Tool Operational Verification and Validation are equivalent to the recurring software activities. In particular, it should be demonstrated that the approach to performing the Tool Operational Verification and Validation activities is equivalent to the low-level requirements based tests, performed on equivalent classes of input files.

7. Supporting Information

Supporting information to help in understanding and applying the tool qualification guidance is provided in DO-330/ED-215 appendixes. This information is either "FAQ" (Frequently Asked Questions: short and concise responses to questions that may be raised) or "DP" (Discussion Paper: clarifications that require more than a short answer). However in DO-330/ED-215 all this information is categorized as "FAQ". But it is important to note that none of these FAQ should be considered as new guidance material.

The FAQ in DO-330/ED-215 are arranged based on their scope:

- Appendix C identifies the FAQ applicable to all domains. These FAQ provides supporting information for Do-330/ED-215 guidance.
- Appendix D identifies FAQ applicable only to Airborne and CNS/ATM software domains. These FAQ mainly refers to Tool Qualification Criteria and Levels, and on certification credit claim for airborne projects.

- ***FAQ C.1: What Does "Protection" Mean for Tools and What Are Some Means to Achieve It?***

  This FAQ provides rationale and examples of protection mechanisms between tool functions. See 5.5- Protection and multi-function tools

- ***FAQ C.2: What Are External Components and How Does One Assess Their Correctness?***

  "External components" is a new topic; this FAQ provides some examples and summarizes the guidance. See 4.7-How to address external components

- ***FAQ C.3: How Can One Maximize Reusability of Tool Qualification Data?***

  Industrial partners would like to benefit from qualified tools without needing to perform additional qualification activities.

  That is not possible, since a tool is qualified only for use on a specific system where the intention to use the tool is stated in the Plan for Software Aspects of Certification. But it is possible to reduce the qualification effort when a tool is used or reused on multiple projects.

  A benefit of DO-330/ED-215 is the work-sharing approach between tool developers and tool users. Based on this separation, the FAQ explains that the tool qualification data may be packaged to maximize its reusability. One suggestion is to consider and adapt the COTS section approach, and/or to separately package the user-dependent and user-independent data.

- ***FAQ D.1: Why Are the Terms "Verification Tool" and "Development Tool" Not Used to Describe Tools that May Be Qualified?***

  This FAQ provides the rationale for one of the main changes in the DO-178C/ED-12C. See Tool Qualification Criteria for Airborne Domain (Chapter 3).

- ***FAQ D.2: Can TQL Be Reduced?***

A note in DO-178B/ED-12B explained the circumstances in which the qualification level may be reduced. The purpose of the FAQ is to replace this note, which was difficult to apply in practice.

The FAQ addresses the same considerations as the note: significance of the certification credit claimed, and the likelihood that other activities would have detected the same errors.

In addition the FAQ emphasizes the need to closely coordinate with the certification authorities and to document the proposed TQL in the PSAC.

It remains to be seen whether this FAQ will be more applicable than the previous note!

- ***FAQ D.3: When Do Target Computer Emulators or Simulators Need to Be Qualified?***

The need to qualify emulators and simulators is often raised in airborne software development projects. The scope of this FAQ is limited to target emulators and simulators used in test environments.

For such tools, qualification may be needed if test cases and procedures are executed in an environment where the target is replaced by an emulator/simulator.

The key consideration is whether the tests performed in such environment are used to satisfy DO-178C/ED-12C objective 6.4.e "The Executable Object Code is compatible with the target computer" (Table A-6 objective 5). This objective references the following activities:

- §6.4.1.a: Selected tests should be performed in the integrated target computer environment, since some errors are only detected in this environment.
- §6.4.3.a: Requirements-Based Hardware/Software Integration Testing

The answer of the FAQ is:

1- If tests are not used to satisfy table A-6 objective 5, then qualification of the emulator/simulator is not required.

2- If tests are used to satisfy (a part of) table A-6 objective 5, then the FAQ considers that the equivalence of the two environments should be demonstrated:

> The differences between the target computer environment and the emulator or simulator environment need to be considered in regard to the ability of tests conducted in the emulator/simulator environment to detect errors typically revealed by the target computer environment testing (see ED-12C/ED-109A section 6.4.1.a) and verify functionalities of the tested software. This could be achieved by analysis (as described in ED-12C/ED-109A section 4.4.3.b) or by emulator or simulator qualification which includes this analysis.

The purpose of the qualification is to demonstrate the equivalence for tests to be executed on an emulator or simulator, and only for these tests. The applicable qualification level is TQL-5. The qualification approach may be based on the execution of a representative set of tests in the two environments and a comparison of the results. Once this is done, then performing the tests only in the emulator/simulator will be allowed. This is of course limited to the tests for which the emulator/simulator is considered as equivalent in term of error detection.

- ***FAQ D.4: What Credit Can Be Granted for Tools Previously Qualified Using DO-178B/DO-278 (ED 12B/ED-109)?***

The FAQ discuss the analyses that are necessary if an applicant proposes to reuse a tool already qualified using DO-178B/ED-12B (or DO-278/ED-109).

The first point to consider is the “verification tool” qualification. When based on DO-178C/ED-12C, if the applicable tool qualification criterion is Criterion 2 instead of Criterion 3, then supplementary qualification effort will obviously be necessary

Compared to DO-178B/ED-12B, the tool qualification guidance is now clarified and accurately specified. Since it is likely that some misinterpretations of the earlier guidance may arise, it is highly recommended that compliance of the actual tool qualification activities be performed with respect to the DO-330/ED-215 objectives.

The FAQ also references section §11.2 (see 5.4- Improvements for previously qualified tools) of DO-330/ED-215, because the guidance provided there should be also considered.

- ***FAQ D.5: What is the Rationale for Tool Qualification Criteria Definition?***

  This FAQ provides the rationale for defining a third tool qualification criterion and includes some examples that show how to determine the applicable criterion. See chapter 3.

- ***FAQ D.6: What are the “Verification Tool” Qualification Improvements?***

  This FAQ summarizes the changes between the guidance for qualifying a verification tool based on DO-178B/ED-12B, and the guidance for TQL-5 in DO-330/ED-215. See 5.2- TQL-5 versus “Verification” tools

- ***FAQ D.7: How Might One Use a Qualified Tool to Verify the Outputs of an Unqualified Tool?***

  This FAQ addresses considerations about tool separation if an applicant proposes qualifying a verification tool to verify the outputs of a non-qualified tool. See 5.5- Protection and multi-function tools.

- ***FAQ D.9: Is Qualification of a Model Simulator Needed?***

  This FAQ is related to the discussion of the use of simulation to satisfy the Model Based Development and Verification Supplement objectives (DO-331/ED-218). It identifies the need to clarify how the tool qualification criteria should be applied to the model simulator.

  In light of a controversy surrounding the claimed benefit of simulation to reduce the required testing, the FAQ only addresses the situation where the certification credit is limited to model verification objectives. In this case the FAQ states that the applicant may propose to not qualify the model simulator.

… for any questions, to ask for a DO-178C/ED-12C or DO-330/215 training, or to propose additional inputs and improvements to this document, please contact :

Frédéric POTHON – ACG SOLUTIONS

(+33) 04.67.60.94.87 – (+33) 06.21.69.26.80

frederic.pothon@acg-solutions.fr

www.acg-solutions.fr