Frédéric Pothon
ACG Solutions

DO-330/ED-215 Benefits of the new Tool Qualification

October 2012

Contributions and Reviews

Laurent Pomies
DOSoft-Consulting

Cyrille Comar
AdaCore

Hervé Delseny
Airbus

Ben Brosgol
AdaCore

Content

1. Purpose of this document

While updating DO-178C/ED-12C, a new document “Software Tools Qualification considerations” was developed. Its cope is both the replace the software tool qualification guidance of DO-178B/ED-12B but also to make possible the use of this “mature” guidance outside the airborne domain.

The purpose of this document is to present how this new document impacts the current tool qualification approach and the how it provides a more relevant guidance for tools.

First this document recalls the rationale for the need for such Tool Qualification document (TQD). To make applicable this TQD, tool qualification criteria and Tool Qualification Levels must be determined. So as an example, this document provides how DO-178C/ED-12C determines these criteria for airborne domain.

Then this document highlights the main impact of this new document on the current practices, and identifies the relevant information to help the reader to apply this new guidance.

Some supporting information is provided in appendix of the TQD. This document emphases the one of the most important, addressing the possible certification credit when using a qualified AutoCode Generator (ACG).

2. Need for a Tool Qualification Document

SC-205/WG-71 considered that it was necessary to develop a clear guidance for qualifying the software tools, to avoid any misinterpretation and difficulties when applying software related guidance to software tools. But also, it seemed necessary to export the tool qualification considerations outside of the “airborne domain”. Therefore a tool vendor might apply a single qualification processes, independently of the domain. The goal is to benefit from a wider tool offer and to increase the tool quality.

For these reason, the concept of “supplement” cannot be applied to “tools”. Then tool qualification considerations are the purpose of a new DO/ED document. The Tool Qualification Document is used in conjunction with the domain related applicable document. To make applicable the Tool Qualification Document, the domain related applicable document should:

- Identify that the Tool Qualification Document is applicable
- Define their own tool qualification criteria
- Define 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, the Tool Qualification Document applies. Therefore, objectives to be satisfied for each TQL are defined, independently of the domain, and of the qualification criteria.

As a first approach, the Tool Qualification Document looks like the 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 also to address all he tool aspects.

The following subsections explain the main principles of the Tool Qualification Document.

3. Tool Qualification Criteria for Airborne Domain

Section 12.2 provides three tool qualification criteria that determine the applicable tool qualification level (TQL) in regard of the software level.

The “criteria 1” addresses the former “development tools”, while the two other criteria split the former “verification tools” depending of the certification credit claim by the qualification of the tool.

Here are the three criteria: DO-178C/ED612C §12.2.2

> 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.

The criteria 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 the tool procedures from the test cases, the certification credit is limited to the correctness of the test procedures (Objectives A7-1).
- The certification credit for a code checker, that verifies the compliance of source code to the coding standard, is limited to the objectives A5-4 Source code is compliant to standard)

The certification credit claimed is extended in case of application of criteria 2 to objectives that are beyond of the data directly verified by the tool.

In appendix of the Tool Qualification Document, a Discussion Paper (DP#5 provides more rationale about the need for these 3 criteria and also some examples of determination between criteria 2 and 3 are provided, using a “proof tool” and a “static ode 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)."

This idea is that the software verification process relies on multiple filters to improve the error detection. With the certification credit claimed in application of criteria 3 is equivalent to remove on filter, considered as useless by the higher level of reliability of the tool. That's way for these tools the Tool Qualification Level (TQL) is higher than for a "classic" verification tool,

The applicable TQL is defined in the table 12-1, based on the qualification criteria and on 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 criteria 1 is the replacement for the development tool for each software level, while the TQL-5 for criteria 3 is the replacement for verification tool in DO-178B/DO-278.

The TQL applicable for Criteria 2 basically requires an increased level of rigor for tools used on software level A and 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 of the compliance of the tool with Tool Requirements. TQL-4 objectives are considered as a minimum to claim confidence in the use of the tool. 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; however, other 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 this document allows an applicant to augment the data in order to satisfy the objectives for the applicable TQL.

The Tool Qualification Document provides some additional information to explain the rationale to not use anymore the terms “development tools” and “’verification tools” FAQ D1. Another FAQ (FAQ D.5) provides rationale to define a third tool qualification criteria and some examples to help the determination of applicable criteria.

4. Principles and Technical aspects

4.1 Domain independent

The goal of this document is to be usable for all domains. However, as a tool may be qualified only in the scope of the "user context" it was difficult to find both a terminology, and to identify the "domain data" that match with all domains.

So it was decided to write the tool document for the airborne software domain, that will be the first and probably main user, and to add a section (§1.3) explaining how to use this document.

Therefore this section explains the need for all domains to define their own tool qualification criteria and tool qualification level, and also that terminology should be adapted for each domain. §1.3:

> 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 the Tool Qualification Document provides an example of 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 of other domain to develop their own tool qualification sections

4.2- Identification of Tool Stakeholders

The scope of the document is to identify all objectives that should be satisfied to qualify a tool in a specific context. So it was important to consider that at least two stakeholder 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 that performed all activities to deliver a tool product to the tool user.

Unfortunately, the direct use of the actors in the process description is not used, but this responsibility separation is identified through:

- section §3.2 that provides 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.
- A new table was provided to identify all objectives (*typically*) applicable to the user

But, there is an exception: In the COTS section (§11.3), the responsibility separation between tool user and tool developer is there explicitly defined.

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 named "Operational Environment" in the document.

> Tool operational environment – The environment and life cycle process context in which the tool is used. It includes workstation, operating system, and external dependencies, for example interfaces to other tools and manual process(es).

Therefore the tool document identifies other environment, used in the framework 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 is the tool verification environment(s) should be representative of the Tool Operational Environment(s).

A the consequence of these definition, there is no anymore some "target" identified in the tool document, but specific objectives were developed to

- Install the tool in the adequate environment (T0-3 for operational environment and T2-8 for verification environment)
- Verify the compatibility of the tool requirements to the operational environment (T3-3)
- And to verify the tool to its operational requirements in the operational environment. (T0-5) and also to perform validation activities 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". But also that it will be used as "system specification".

The Tool Qualification Document clarifies the different steps of requirements, starting with the "Tool Operational requirements" that is the software life cycle needs. So it is equivalent to the system specification for software. But its content 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 in one or several level of "Tool requirements", poorly identified as "Tool requirements" and "tool low-level requirements". Each refinement level may include some derived requirements. Those derived requirements are those that are not traceable to the higher level (simpler definition that 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 may not document all tool functions, but only those required by the user. This is not the case for the Tool Requirements that need to describe all tool functions and features. These extraneous functions will be then identifies as derived requirements, and then analysed.

4.5- Need for Tool Validation

Software Requirements Validation is out of the scope of DO6178C/ED-12C, it is under the responsibility of the system processes. To make a parallel with the tool, the use of the tool in the software life cycle processes will allow to assess that the tool is compliant with the user needs, defined or not, in the Tool Operational Requirements.

So the term validation was included in the tool document and is the purpose of two complementary objectives:

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

§6.2.1:

> 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 supplements the verification objectives performed on the Tool Operational requirements and on the Tool itself for compliance to the Tool Operational requirements, That's why the tool document identifies in §6.2 the objectives and activities of the "Tool Operational Verification and validation Process".

4.6- A New Table for User Objectives

10 objective tables in DO-178C/ED-12C, but 11 in the Tool Qualification Document! Here is this 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 process. So this table (for "Tool Operational Processes) identifies objectives on

- Planning process: To define the need for qualification and the applicable tool qualification level. Typically these information are 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 4 objectives of the Tool Operation verification and Validation process.

4.7-How to address External components

The application of DO-178B/ED-12B to development tool for software level raised concerns on the object code to source code traceability aspects.

This is removed from the Tool Qualification Document. But additional considerations on "external components" were added.

First these external components are defined in the glossary as:

> 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 appendix of the Tool Qualification Document.

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

Then the tool requirements are verified. This verification includes the completeness and consistency of the requirements to address the failure modes.

Finally the objectives "Tool Executable Object Code is robust with Tool requirements/Tool low-level requirements" are satisfied by developing test cases form the Tool requirements (and Low-level if any) identifying the failure modes.

In addition, it was also agreed that a general behaviour may be defined, without identifying specific failure modes. In such a case, some additional test cases should be developed to complete the demonstration of capability of the tool to answer to abnormal conditions or data. Here is the corresponding text (§6.1.4.2) about Tool testing activities:

> 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?

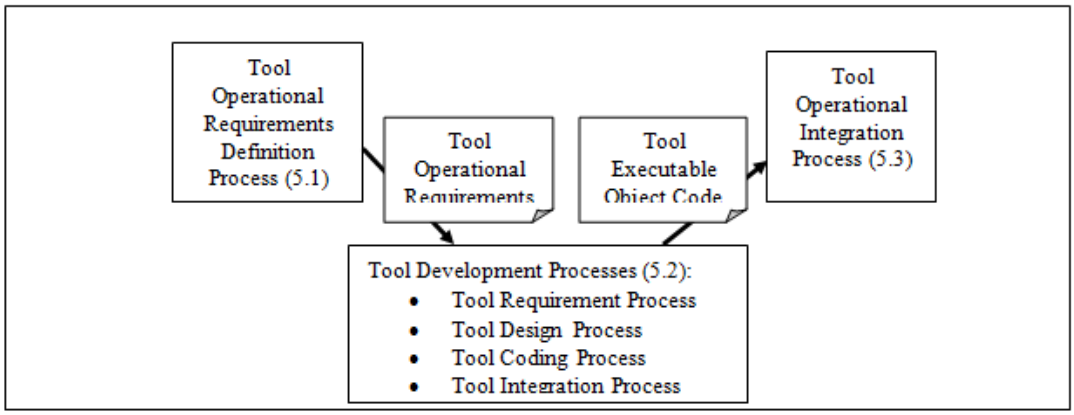
5.1- Tool user and tool developer processes

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

- Planning process,
  - Tool User: it is recalled that the user should identify the need and level of qualification for the tool and rationale in term of certification credit claimed. For that purpose, the document identifies in consistency with DO-18C/ED-12C the specific information to be provided in the PSAC. Even if it is well known in airborne domain, as the document is domain independent, it was relay important to identify to new comers that a similar approach is necessary for all domains. In addition the PSAC should also described (or references) the description of processes to be performed by the user.
  - Tool Developer: All objectives of the planning process are applicable, but limited to its perimeter.

- 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 be the user, in the Tool Operational Requirements. This is the "*Tool Operational Requirements Definition Process*.
  - Tool Developer: The tool is developed form the Tool Operational Requirements in compliance with the Tool Life cycle defined in the Tool Development Plan. This is typically based on specification, design, coding, and integration process. 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 install 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 similarly to those provided in DO-178C/ED-12C: verification of output of 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.
- 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 to its operational environment is addressed by tool user activities. This approach will normally facilitate the qualification renewing in case of Too Operational Environment changes (e.g. upgrade of workstation).

- SQA and SCM process: There is no objective separation in the document for these processes. However, as the planning, development and verification process are composed of complementary activities, to satisfy the SCM and SQA objectives, an organization should be setup to manage and oversight the complete life cycle processes, both under the framework of the tool user and the tool developer.

- Qualification liaison process: The objectives of the Tool Qualification Liaison process is based on the complementary data provided both by the tool developer and the tool user. The data provided should address the complete life cycle processes, whatever the packaging is.

5.2- TQL-5 versus "Verification" tools

As defined in 4.5.5, TQL-5 is equivalent to the qualification level for "verification tools" in DO-178B/ED-12B. The initial intend was to keep the same level of rigor for these tools to not prevent their use (so not raise the bar). So, the objectives applicable to TQL-5 should be normally only the implementation of the qualification criteria of DO-178B/ED-12B: "the tool complies with its Tool Operational Requirements under normal operational conditions".

But also, "software configuration management process and software quality assurance process objectives should apply"

The Tool Qualification Document 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 (e.g., the qualification does not require any data from the tool development process). However, it clarifies the content of the TOR, the compliance of the tool to the resulting software process needs, and the objectives of other integral processes applicable for TQL-5.

So the objectives applicable to 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. More the validation objectives create a tie between the Tor and the certification credit claimed. Evidences 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 to TQL-5.: Identification of configuration items and archive for 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 Qualification Liaison process, applicable to all levels, is to analyse the known problems for possible impact on the Tool Operational Requirements. It is a little contradictory with the possible absence of data from tool vendor. But is seems to the committee that this analysis should conducted to identify possible limitations of the tools, that may reduce the certification credit claimed.

Details on “Verification Tool” Qualification Improvements” is the purpose of a FAQ in appendix of the Tool Qualification Document. (FAQ D.6)

5.3- A convenient approach for COTS tools

One of the skates of the Tool Qualification Document was to facilitate and to clarify the qualification of commercial tools.

We can say that this is achieved, first by the definition of stakeholder, and by the definition of complementary processes between tool user and tool developer.

So the section §11.3 of the Tool Qualification Document develop a possible way to satisfy the tool qualification objectives in case of COTS tool. The main problem when trying to apply the tool qualification guidance is that the tool is NOT developed form Tool Operational Requirements develop by a user.

So the section §11.3 proposed to split the TOR content in two parts:

- A developer-TOR that is use to develop the tool. It is also used for verifying the compliance and traceability of Tool requirements. Similarly the developer also provides a developer-TQP, developer-TCI and developer-TAS limited to its activities.
- The developer data are supplemented by other data provided by the user: The TOR that includes or references the developer-TOR and TQP, TAS, TCI . This TOR is used for the Tool Operational Verification and Validation process.

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 QUQLIFICATION LIAISON PROCESS			
All	TOOL USER		

5.4- Improvements for Previously qualified tools

In the “additional considerations section of the Tool Qualification Document the ruse aspect is addressed through “Previously Qualified Tools” section (§11.2)

The sections below provide guidance for three aspects of tool qualification:

1- Reuse of previously qualified tools that are unchanged

In this paragraph, the document provides criteria to be analysis to be sure that the tool is suitable for reuse without any change. The criteria includes 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 limited to the representativeness of the tool verification environment, and on the tool operational verification and validation process. 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 does “partitioning” means for tools? It was considered that this concept was not directly applicable for tools. But is may be sometimes necessary to guarantee a form of isolation between tools or between tool functions to prevent the presence of common errors.

After a lot of discussion, the term “protection” was used and defined in a glossary as:

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

The concept of this protection is applicable when a different level of qualification is proposed for several 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.

To know more how to understand and to implement the protection mechanism, the Tool Qualification Document includes a FAQ (FQA C.1 “*Does “Protection” Mean for Tools and What Are Some Means to Achieve It?”*

First, this FAQ extends the application of protection between two tools (not only between tool functions).

Then, it lists without pretending being exhaustive of course, some possible techniques

- partitioned spatially and temporally
- functional partitioning strategy
- functional deactivation techniques

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

Based on this concept, there are two possible applications:

1- **Multi-function tools** is the purpose of the section 11.1 of Tool Qualification Document. This section is applicable when one proposes to qualify only some functions of the tool, or not all functions at the same level. This approach is possible only if 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 not is added that the guidance of this section, including protection, is not applicable, when "*the tool does not contain functionality above TQL-5*."

2- **Verification of the outputs of a non-qualified tool**: It could be a multi-function tool that both produces an output and verifies this same output, or several tools. In this case, 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 is required with independence, a higher degree of protection (call "independence between tools) will be required.

For this second application, the FAQ D-7 was created to address a lot of concerns about this approach. The main concern was about the ability of the qualified tools to satisfy all objectives applicable to the outputs of the non-qualified tool.

This FAQ addresses the following considerations regarding the possible problems raised when using a tool(s) to verify the outputs of an unqualified tool:

- Coverage of verification objectives that apply to the unqualified tool's output that is the main concern.
- 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 team, separate tool development process and/or dissimilar technical approach. The FAQ insists also on the problem to use common components (such as libraries) in both tools
- Protection between the tools (that is, avoiding interference of the unqualified tool on qualified tool's proper operation).

5.6- Use of Service History to qualify a tool

The guidance of the Tool Qualification Document 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 done in DO-178C/ED-12C, it might be still very difficult to claim credit with this "alternate mean" for software.

For tool, that's could be different.

Section 11.4 explains that the reason to use service history, among other, may be the need to increase the TQL. This could be very interesting for criteria 2 tool that requires for software level A and B the application of TQL-4 instead of TQL-5 for level C and D or for

criteria 3. But the difference between the 2 criteria is only based on the certification credit claimed through the qualification of the tool, not on the tool functions.

So an acceptable approach may be first to qualify a tool in application of criteria 3 a tool (then with the limited certification credit), so at TQL-5. So in the scope on operational use, the tool service history may be recorded. This service history includes that some verification activities never detect any errors. Based on these evidences, it could be possible to propose to increase the certification credit of the tool based on the alleviation of useless verification activities. This corresponds to qualify the tool at TQL-4 due to the application of criteria 2 (for software level A or B).

This qualification at level TQL-4 will be based on qualification at TQL-5 with additional data from service history.

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

First it seems strange to need to qualify a tool for qualifying a tool. But if you look more accurately, it is a good practice to automate some activities to 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 “development tool” to be qualified.

In DO-178B/ED-12B the approach is required due to the “recursively” of the tool guidance: To qualify a development tool, the same objectives than the software should be satisfied … including additional considerations, so the tool qualification section.

In DO-178C/ED-12C, the tool qualification section once identifies the need for qualification and the applicable TQL, and then refers to the Tool Qualification Document. So a specific guidance is provided in the Tool Qualification Document to address the need for tool qualification for the tool used in the Tool life cycle.

Tool planning objective on the need to address additional considerations includes explicitly the assessment of “the need to to qualify any tool(s) used in the framework of the tool life cycle processes”

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

- Tool that may inject an error to be qualified at the same level than the tool to be qualified itself
- Tool that main only fail to detect an error 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 the criteria 2 is not applicable to the “second layer” tools!

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

DO-178C/ED-12C text supplements explain that one supplement may be used in conjunction with any other supplements. But nothing similar in the Tool Qualification Document! Therefore is allowed to use the DO-178C/ED-12C supplement to qualify a tool?

The Tool Qualification Document is domain independent. To make this document applicable, a domain dependent document, such as DO-178C/ED-12C, should reference the Tool Qualification Document. But this independency will be violated if the Tool Qualification Document references the DO-178C/ED-12C supplements.

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

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

So to use a supplement for qualifying a tool, it is necessary during the Tool Qualification planning process

- To review all potentially relevant supplements and identify those that will be used,
- To identify the impact of the use of the selected supplement on the tool qualification objectives to be satisfied.
- To document in the TQP the mean to satisfy all the objectives, as adapted by the supplement where applicable

6. Certification Credit for a Qualified ACG

Nowadays, almost 10% of the criteria1 tools are AutoCode Generator (ACG) including database, parameter data and similar. The qualification of a these tools requires a lot of workload, similar than the software itself. But it is not clear what possible certification credit that may be claimed when using this kind of tool.

After very tough discussion, a Discussion paper was approved on this topic!

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 process 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 cut off a legend: A code generator should not be used to verify the model: The purpose of a code generator is to translate a model into sour 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*")

  First, the "correctness and completeness" of the translation is mainly based on the accuracy of the tool requirements and on the 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 tool verification activities are complete and relevant to the Source Code verification objectives, credit may be claimed for 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 required 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).

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

  - Scenario 1 - Satisfaction of low-level requirements-based tests objectives through tests cases based on the low-level requirements.
  - Scenario 2 - Satisfaction of low-level requirements-based tests objectives through tests 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 it is identified if the objectives is satisfied through tool qualification process ("tools" or through airborne software processes ("Software") or the coverage involved 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 of these analyses 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). It is the Tool Operational Verification and Validation that should be considered as equivalent as the recurring software activities.

7. Supporting Information

FAQ and DP in the Tool Qualification Document are sorted depending of their scope:

- Appendix C identifies the FAQ/DP applicable to all domains
- Appendix D identifies FAQ and DP applicable only to Airborne and CNS/ATM software domains

- ***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 mechanism between tool functions. See Protection and multi-function tools

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

  As "external components" is a new topic, this FAQ proposed some examples and summarizes the guidance. See External components

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

  Industrial partners would like to benefit from qualified tools without any need for additional qualification activities.

  That is not possible as 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 what it is possible is to reduce the qualification effort when a tool is use or reuse on multiple projects.

  The benefit of this new Tool Qualification Document is the work-sharing approach between tool developers and tool users. So based on this separation, the FAQ explains that the tool qualification data packaging may be defined to maximize the reusability. It is for example suggested to consider and adapt the COTS section approach and/or to package separately the data that is user-defendant from data that 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 change in the DO-178C/ED-12C. See Tool Qualification Criteria for Airborne Domain.

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

  In DO-178B/ED-12B, a note existed providing upon which considerations the qualification level may be reduce. The purpose of the FAQ is to replace this note that was really difficult to apply.

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

  In addition the FAQ insist on the need to closely coordinate with the certification authorities and documented in the PSAC the proposed TQL.

  Not sure that this FAQ will be more applicable that the previous note!

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

Need to qualify emulators and simulators raised very often in the projects. The scope of this FAQ is limited to target emulators simulators used in test environments.

For such tools, the need for qualification is dependent of test cases and procedures executed in the environment where the target is replaced by an emulator/simulator.

The key aspect is to determine if the tests performed in such environment are used or not to satisfy the objective 6.4.e "The Executable Object Code is compatible with the target computer." (Table A-6 objective 5). This objective references

- §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

So the answer of the FAQ is:

1- Tests are not used to satisfy table A-6 objective 5: The qualification of the emulator/simulator is not required.

2- Tests used to satisfy (a part) table A-6 objective 5: Analysis of 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. This could achieve by analysis or by emulator/simulator qualification."*

The purpose of the qualification is to demonstrate the equivalence for tests to be executed on an emulator or simulator, and only for these ones. 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 to compare. Once it is done, then only execution in the emulator/simulator will be allowed for the test 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 ED 12B/ED-109?***

  The FAQ discuss the necessary analyses to be conducted if an applicant proposes to reuse a tool already qualified using DO-178B/ED-12B.

  The first aspect to consider is the "verification tool" qualification. When based on DO-178C/ED-12C, the applicable tool qualification criteria is criteria 2 instead of criteria 3, it will be obvious that a supplementary qualification effort will be necessary.

  As the objectives to be satisfied in the Tool Qualification Document are clarified, it is possible that the interpretation done in application of DO-178B/ED-12B is sometime deficient. So analysis of these differences should be conducted.

  The FAQ references also the section §11.2 (see Previously qualified tools) of the Tool Qualification Document, because guidance provided there should be also considered.

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

  This FAQ rationale to define a third tool qualification criteria and some examples to help the determination of applicable criteria. See Tool Qualification Criteria for Airborne Domain

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

  This FAQ summarizes the changes between the guidance to qualify a verification tool based on DO-178B/ED-12B, and the one applicable to TQL-5 based on the Tool qualification document. See 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 it is proposed to qualify some verification tools to verify the outputs of a non-qualified tool. See Protection and multi-function tools.

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

  Related to the discussion on the use of simulation to satisfy the Model Based Development and Verification supplement objectives, it raised the need to clarify how the tool qualification criteria should be applied on the model simulator.

  Due to controversial discussion on the claim benefit of simulation to alleviate the tests, the FAQ only addresses when the certification credit is limited to model verification objectives In this case the FAQ answers 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