Qualifying Software Tools According to ISO 26262

Mirko Conrad[1], Patrick Munier[2], Frank Rauch[3]

[1] The MathWorks, Inc.,
Natick, MA, USA
mirko.conrad@mathworks.com

[2] The MathWorks, SAS,
Grenoble, France
patrick.munier@mathworks.fr

[3] TÜV SÜD Automotive GmbH
Munich, Germany
frank.rauch@tuev-sued.de

Abstract:

The growing adoption of safety standards in the automotive industry results in an increasing interest in as well as an increasing uncertainty about software tool certification and qualification. With ISO 26262 on the horizon, new tool qualification requirements need to be understood and implemented by automotive software practitioners.

This paper summarizes the tool qualification approach of ISO/DIS 26262 and contrasts it with tool certification and qualification requirements outlined in other safety standards and guidelines. The authors also report about their first-hand experiences with qualifying development and verification tools according to ISO/DIS 26262 in practice.

1 Tool Certification / Qualification Approaches in Standards and Guidelines

This section is intended to provide an overview about the requirements in popular safety standards and guidelines pertaining to qualifying or certifying software tools. The following discussion should provide the context for a more detailed discussion of the ISO/DIS 26262 tool qualification approach in sections 2 and 3.

So far, there is no single approach for tool qualification or certification across standards. Rather, different standards attach different levels of importance to tool certification / qualification and suggest different approaches to gain confidence in the tools used.

Typically, tool users are responsible in the end for the certifying or qualifying the software tools they are using. Tool vendors can support these efforts by providing certification or qualification kits that ease the certification or qualification efforts on the user's side.

The safety standards and guidelines discussed in the following paragraphs target different application sectors with domain-specific requirements. The amount, scope, complexity and criticality of software tools used during the development of high-integrity systems may differ between these sectors. From the authors' point of view, this might be one of the reasons for having divergent tool qualification / certification requirements.

1.1 DO-178B

DO-178B is considered to be one of the most stringent safety standards used in industry. It clearly defines the conditions under which a software tool needs to be qualified (DO-178B,

section 12.2), i.e. tool qualification is required if the answer to each of the following three questions is 'yes':

(1) Can the tool insert an error into the airborne software or fail to detect an existing error in the software within the scope of its intended usage?
(2) Will the output of the tool not be verified as specified in Section 6 of DO-178B?
(3) Will the output from the tool be used to meet or replace an objective of DO-178B, Annex A?

Tool qualification is defined as the "process necessary to obtain *certification credit*[1] for a software tool within the context of a specific airborne system" (DO-178B, Glossary)

There are two types of tools that may require qualification: development and verification tools. If the answer to the question:

(4) Is the tool output part of the airborne software, such that the output can insert an error into the software?

is 'yes' the tool needs to be qualified as development tool, otherwise it needs to be qualified as verification tool. Verification tools are simpler to qualify, because they cannot, by definition, inject errors into the final product.

Development tool qualification requires that the development process for the tool satisfies the same objectives as the software development process of the airborne software. The software level assigned to the tool is the same as for the airborne software it produces unless the applicant can justify a reduction of the tool's software level to the certification authorities.

Verification tool qualification requires the user to demonstrate that the tool complies with its operational requirements under normal operation conditions. In practice this can be achieved with documented tool operational requirements and a set of test cases that allow one to verify, that the tool correctly implements these requirements under normal operating conditions.

Tools need to be qualified on a per project basis. Tool qualification activities are subject to a planning phase that is documented in a tool qualification plan. Using qualified tools allows one to achieve dedicated certification credits.

DO-178B tool qualification is widely discussed e.g. in [HB07, KZ09].

1.2 IEC 61508

IEC 61508 (Ed. 1.0) includes the concept of *tool certification*. The standard recommends (for SIL 1) or highly recommends (for SIL 2-4) the use of *certified tools and translators* (IEC 61508-3, table A.3). Wherever possible, tools should be certified so that some level of confidence can be assumed regarding the correctness of their outputs (IEC 61508-7, clause C4.3.). As an alternative, tools with *increased confidence from use* are highly recommended for all SILs.

The standard provides little guidance on how to certify a tool. As a result a variety of different tool certification approaches has been used in practice [KM03, JG03, Glö08, TMW08, TMW09, SLM09, BB09]. Developing a software tool as per IEC 61508-3 is just one of multiple options here.

Tool certification is viewed as a generic measure to increase the confidence in the tools used, if increased confidence from use cannot be claimed. No certification credits are offered in

[1] Acceptance by the certification authority that using a tool satisfies a certification requirement

exchange for tool certification. This way there is little incentive for applicants to certify tools if it can be avoided. Requirements for tool qualification are limited, because the IEC 61508 software life cycle for embedded software is set-up to uncover potential tool errors.

IEC 61508-3 also emphasizes the use of an integrated tool chain (IEC 61508-3, clause 7.4.4.2).

The upcoming second edition *IEC 61508 (Ed. 2.0)* distinguishes between online[2] and offline[3] tools. *Software offline support tools* are divided into the following categories (cf. IEC 61508-4 Ed. 2.0, clause 3.2.11):

- T1: tools that cannot directly or indirectly contribute to the executable code of the system
- T2: tools supporting verification or test of the design or the executable code; errors in these tools can fail to reveal defects
- T3: tools generating outputs that can directly or indirectly contribute to the executable code of the system

Depending on the tool's classification, the identification of use cases and related hazards is required. T3 offline tools require evidence that the tool conforms to its specification or manual. This evidence may be based on confidence from use or on application independent validation. If such evidence is not available, application-specific measures to uncover potential tool errors, such as back-to-back testing, could be implemented by the applicant.

Each new version of an off-line support tool shall be qualified. This delta qualification may rely on evidence provided for an earlier version.

Both editions of IEC 61508 very much value confidence from use as a measure to gain confidence in the tools used.

1.3 ISO/DIS 26262

Although ISO/DIS 26262 is considered a derivative standard of IEC 61508, tool qualification approaches in these two standards differ significantly.

ISO/DIS 26262 contains detailed guidance on *software tool qualification* (ISO/DIS 26262-8, 11). The objective of tool qualification is to provide evidence that a software tool is suitable for use in the development of safety-related software according to ISO/DIS 26262.

The use cases for a tool need to be documented and analyzed. This analysis shall evaluate if the malfunctioning software tool or its erroneous output can lead to the violation of a safety requirement. In addition, the probability of preventing or detecting such errors in the output of the tool needs to be evaluated. Based on this analysis, a required *tool confidence level* (TCL) is determined.

The required TCL, together with the Automotive Safety Integrity Level (ASIL) of the safety-related software developed using the software tool, guides the selection of the appropriate tool qualification methods. These tool qualification methods are listed in tables 2, 3, and 4 of ISO/DIS 26262-8.

Increased confidence from use is one of the possible qualification methods, however the level of recommendation in the above mentioned tables decreases for the higher TCLs and ASILs.

---

[2] Software on-line support tool: software tool that can directly influence the safety-related embedded system during its run time

[3] Software off-line support tool: software tool that supports a phase of the software development life cycle; cannot directly influence the safety-related embedded system during its run time

With this approach, ISO/DIS26262 provides very detailed guidance to classify the tools and to estimate the required level of tool qualification. However, unlike earlier draft versions, ISO/DIS 26262 does not distinguish between different tool categories.

Details of the ISO/DIS 26262 tool qualification will be discussed in chapters 2 and 3. [Sau09] touches upon this topic as well.

1.4 MISRA-C

MISRA-C sees compilers and static checking tools as *trusted processes*. This means that there is certain level of reliance on the output of the tools. It needs to be ensured that this trust is not misplaced (MISRA-C:2004, section 4.2.3).

MISRA-C:2004 suggests documented *validation testing* as the method of choice to gain confidence in the tools used.

Ideally, the tool supplier should carry out the validation tests. The tool vendor should be able to provide records of the verification activities and change records to document a controlled software tool development. The tool vendor should also have a bug reporting and tracking mechanism. The size of the user base together with an assessment of the bugs reported over the last 6-12 months is seen as an indication of the stability of the tool.

If tool validation information is not available from the tool vendor, confidence in the tool can be gained by the tool user for example by adopting the following approaches:

- Documented validation testing
- Assessment of the development processes use by the tool supplier
- Review of the tool's performance

2 The ISO/DIS 26262 Tool Qualification Approach

This section provides a brief overview about the tool qualification approach as outlined in the draft standard.

2.1 ISO/DIS 26262

ISO 26262 is an emerging international safety standard titled *Road vehicles — Functional safety*. This sector specific standard for the automotive industry is intended to be applied to safety-related systems that include one or more E/E systems[4], and are installed in series production passenger cars with a maximum gross weight of up to 3.5 tons.

The draft international standard, ISO/DIS 26262 was published in summer 2009. Part 8 of the draft international standard, ISO/DIS 26262-8 *Supporting processes*, addresses multiple cross-functional topics, including the *qualification of software tools*.

2.2 Tool Qualification Overview

The use of software tools can simplify or automate activities and tasks required by ISO 26262 for the development of safety-related software (ISO/DIS 26262-8, 11.2). The objective of ISO 26262 software tool qualification is to provide evidence that a software tool is suitable for use

[4] Systems that consists of electrical and electronic elements, including: programmable electronic elements, power supplies, input devices, communication paths, and output devices.

in the development of safety-related software, such that confidence can be achieved in the correct execution of activities and tasks required by ISO 26262 (ISO/DIS 26262-8, 11.1).

To determine the required tool confidence level (TCL), the use cases of the tool shall be analyzed. This analysis shall evaluate:

- If a malfunctioning software tool and its erroneous output can lead to the violation of any safety requirement allocated to the safety-related software to be developed.
- The probability of preventing or detecting such errors in the output of the tool.

The evaluation considers measures internal to the software tool (for example, monitoring), as well as measures external to the software tool (for example, guidelines, tests, and reviews) that are implemented in the development process for the safety-related software.

The required TCL, together with the Automotive Safety Integrity Level (ASIL) of the safety-related software developed using the software tool, allows the selection of the appropriate tool qualification methods. Tool qualification can be carried out for individual tools as well as for tool chains or sets of tools.

The ISO/DIS 26262 tool qualification process requires the creation of the following *tool qualification work products* (ISO/DIS 26262-8, 11.5; see the appendix for a summary):

- Software Tool Qualification Plan
- Software Tool Documentation
- Software Tool Classification Analysis
- Software Tool Qualification Report

2.3 Software Tool Qualification Plan (STQP)

The software tool qualification plan is a planning document created in an early phase of the development of the safety-related system.

Besides stating the applicant, and the application under consideration, it identifies the tool and tool version to be qualified, the intended configuration and operational environment. In this sense, the STQP shares conceptual similarities with tool qualification plans used in DO-178B projects.

The tool qualification plan also lists the intended tool use cases. It is supposed to list the tool qualification methods and available means to detect malfunctions or erroneous output of the tool.

2.4 Software Tool Documentation (STD)

The software tool documentation comprises different information that the tool applicant may need when using the tool. It comprises information such as: tool overview, available tool documentation set, operational environment and constraints, installation instructions, known issues.

The STD also provides information necessary to check whether the use cases, configurations and operational environment listed in the STQP are supported by the tool. It has similarities to the description of tool operational requirements as per DO-178B.

2.5 Software Tool Classification Analysis (STCA)

The tool classification is necessary to determine the required *tool confidence level* (TCL). It depends on the particular tool use cases used during the development of the application under consideration. The tool confidence level can be derived using the schematics provided in Fig. 1.

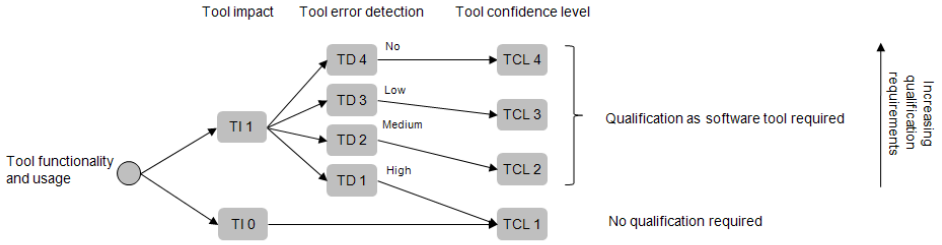

Fig.1: ISO/DIS 26262 Tool Classification Scheme

Tool Impact (TI)

First, the intended use cases for the tool shall be analyzed to determine if a safety requirement can be violated if the software tool is malfunctioning or producing erroneous output. If a violation of a safety requirement is not possible, *tool impact* TI0 shall be chosen. Otherwise, the tool impact is TI1 (ISO/DIS 26262-8, 11.4.3.2.a).

Tool Error Detection (TD)

Second, the intended software tool use cases shall be analyzed to determine the probability of preventing or detecting that the software tool is malfunctioning or producing erroneous output. The degree of confidence, that a malfunction or an erroneous output from the tool can be prevented or detected, determines *the tool error detection* TD as outlined in Table 1 (ISO/DIS 26262-8, 11.4.3.2.b).

Tab.1: Tool Error Detection Levels

Degree of confidence	Tool error detection
high	TD1
medium	TD2
low	TD3
no systematic verification measures in subsequent development phases malfunctions or erroneous outputs can only be detected randomly	TD4

Tool Confidence Level (TCL)

If TI and TD have been chosen, the *tool confidence level* (TCL) can be determined following the schematics provided in Figure 1 (ISO 26262-8, 11.4.3.4).

Having multiple use cases for a tool can potentially result in multiple TCLs. To determine the required tool qualification measures, the maximum TCL required (TCL_{REQ}) to support these use cases needs to be established. TCL_{REQ} needs to be documented in the STQP.

Tool Qualification Methods (TCM)

A tool classified at TCL1 does not require specific *tool qualification methods* to be carried out. For software tools classified at any of the other tool confidence levels, specific methods for software tool qualification shall be applied. These specific methods are listed in ISO/DIS 26262-8, tables 2, 3, and 4 and have ASIL specific recommendations. From the authors' perspective, some of these methods seem to be feasible for commercial off the shelf tools only.

As an example, permissible tool qualification methods for TCL2 are given in Table 2. To qualify a tool classified at TCL2 up to ASIL D, methods 1b, 1c, 1d or a suitable combination of these could be used.

Tab. 2: Tool Qualification Methods for TCL2

Method	ASIL			
	A	B	C	D
1a Increased confidence from use	++	++	++	++
1b Evaluation of the development process	++	++	++	++
1c Validation of the software tool	+	+	+	+
1d Development in compliance with a safety standard	+	+	+	+

(+ … method recommended; ++ … method highly recommended)

The selected tool qualification methods to be carried out need to be documented in the STQP.

2.6 Software Tool Qualification Report (STQR)

The software tool qualification reports documents the actual tool qualification, i.e. provides evidence that the tool qualification was carried out as planned. Usage constraints and malfunctions identified during the qualification, if any, shall be documented here as well.

3 Experiences with tool qualification according to ISO/DIS 26262 - A practitioner's perspective

MathWorks automotive industry customers have expressed their need for compliance with the upcoming ISO 26262 standard [TMW09] and for tools qualified as per ISO 26262 in particular. In order to support this customer need, a practicable ISO 26262 tool qualification approach had to be developed and implemented.

In this section the authors discuss their first hand experiences with the ISO/DIS 26262 tool qualification approach gained during the qualification of the Real-Time Workshop® Embedded Coder™ code generator and the PolySpace® Client/Server for C/C+ code verifiers from MathWorks. These qualification activities happened in 2009.

3.1 Implementation of the ISO 26262 tool qualification approach

ISO 26262 allows different levels of qualification, including a self-qualification (1st party qualification). To increase confidence into the proposed approach, MathWorks decided to submit the tool qualification approach to an accredited certification body for review and approval. Due to their reputation for software tool certifications / qualifications according to

various standards, TÜV SÜD Automotive GmbH was chosen for the tool qualification assessment.

Due to their large customer base, MathWorks supports customers developing high-integrity systems according to different standards. Tool certification packages for IEC 61508 and qualification kits for DO-178B were already productized when the ISO 26262 qualification activities were launched. So it was desirable to re-use certification / qualification approaches and artifacts developed for these other standards wherever feasible.

A suitable way to leverage existing certification approaches was to add the ISO 26262 tool qualification on top of the existing IEC 61508 in-context tool certifications. These existing certifications are based on specific workflows (*reference workflows*) to be utilized by the applicant when using the tool for developing or verifying software for IEC 61508 applications. In the context of ISO 26262 tool qualification, these workflows can be re-used to describe and limit the intended tool use cases as well as to list available means to detect malfunctions or erroneous output of the tool.

In the following, we will illustrate this approach using the Real-Time Workshop Embedded Coder code generator as an example:

The reference workflow describes a workflow for application-specific verification and validation of models and generated code developed using the Simulink® modeling environment and the Real-Time Workshop Embedded Coder C code generator. The main constructive development activities as well as the corresponding verification and validation activities are summarized in Fig. 2. [Con09, CS09] provide more detailed discussions.

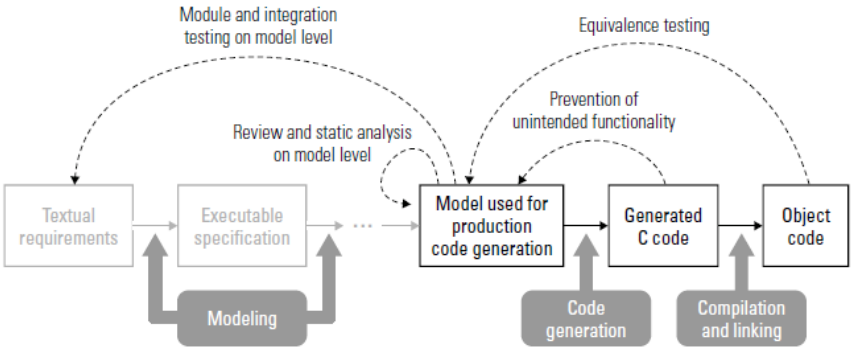

Fig.2: Reference Workflow for Application Specific Verification and Validation of Models and Generated Code

The verification and validation measures described in this reference workflow form available means to detect or prevent potential malfunctions or erroneous outputs of the code generator. This description was used to determine the tool error detection (TD) for the code generator.

According to the certification report, applying the entire workflow for Application-Specific Verification and Validation of Models and Generated Code provides a high degree of confidence that potential malfunctions or erroneous output of the code generator can be detected or prevented, i.e. the tool error detection is TD1 resulting in a tool confidence level of TCL1. Tool qualification for the code generator can be claimed without carrying out additional tool qualification methods.

To give the user more flexibility when integrating Real-Time Workshop Embedded Coder into their development processes MathWorks also aimed at supporting use cases that only utilize a suitable subset of the reference workflow[5]. Assuming that the selected subset provides a medium degree of confidence to detect or prevent potential malfunctions or erroneous output of the code generator, the tool error detection is TD2 resulting in a tool confidence level of TCL2. In this case, tool qualification methods need to be selected according to Table. 2.

The PolySpace verifiers for C/C++ were classified at TCL2 as well.

In case of both products, a combination of the tool qualification methods (1b) Evaluation of the development process and (1c) Validation of the software tool were used.

The *assessment of the development process* was carried out by TÜV SÜD as part of the IEC 61508 tool certification procedure. The assessment criteria were enhanced to match the ISO 26262 requirements.

The *software tool validation* was carried out differently for the two tools. In case of PolySpace for example, existing test artifacts that were created as part of the DO-178B tool qualification kit for PolySpace were reused.

For both tools, the tool classification reports, artifacts to demonstrate compliance with the documented development processes and evidence for the validation of the tools were submitted to TÜV SÜD for review. TÜV SÜD assessed the artifacts and stated their suitability to claim tool qualification in the certification reports for Real-Time Workshop Embedded Coder and PolySpace.

To further support users of Real-Time Workshop Embedded Coder when claiming tool qualification, a *Tool Qualification Package* was created. The package contains templates for the tool qualification work products identified by ISO/DIS 26262-8 as well as evidence documenting the independent assessment by TÜV SÜD of the tool qualification measures carried out by MathWorks. In the tool qualification package, the tool qualification work products are integrated into one single document to account for the overlap and dependencies between the different work products (see section 4 for a detailed discussion of this issue).

3.2. Discussion

The authors see the following issues with the tool qualification approach outlined in ISO/DIS 26262:

No formal qualification credits

Similar to IEC 61508, ISO/DIS 26262 has a generic requirement for tool qualification, but doesn't offer formal certification credits in exchange for tool qualification. This way there is little incentive for applicants to certify tools if it can be avoided. As such, it is feared that the tool classification will be dressed up to reduce or avoid tool qualification methods.

Assuming a proper tool classification and qualification, one could argue that a qualified tool used in accordance with an appropriate reference workflow might provide the required confidence in the correct execution of activities automated by this tool. Following this argument, the verification activities for these automated activities defined in the standard can be reduced or eliminated. Since there is no straightforward mapping between activities/tools

[5] The applicant needs to document the chosen workflow subset in a conformance demonstration template that ships with the tool qualification package.

and their corresponding verification activities, this will be open to interpretation until common practices have been established.

No distinction between verification and development tools

Unlike DO-178B and IEC 61508 Ed. 2.0, ISO/DIS 26262 does not differentiate between tools that can introduce errors (aka development tools) and tools that can only fail to detect errors (aka verification tools). Imposing the same tool qualification requirements for both classes of tools does not account for the different criticalities of these tool categories. Providing a generic reduction of the TCL for verification tools could be one means of addressing this issue. If this is not addressed properly in new versions of the standard, the authors are concerned that the tool categories are treated differently when assigning a TD level. However, this would be less transparent than a clear statement in the standard itself.

Re-using the same arguments for several tools

A single means to detect or prevent errors in a tool could be used in the argumentation to lower the tool confidence level for several tools. This seems to result in a lower confidence for the overall tool chain, when compared with using different means to lower the confidence levels of different tools. However, ISO/DIS 26262 does not seem to provide any guidance on how to deal with these cases of 'double accounting'. It would be helpful to have some kind of 'TD/TCL arithmetic' that allows the determination of tool confidence and tool error detection levels when combining tools or tool chains that have been classified already.

Difficulty to provide a reasonable tool classification without considering the entire tool chain

The above-mentioned issue also leads to the problem that proper tool classifications and qualifications are difficult to achieve without considering the entire tool chain. This raises questions of how feasible the qualification of individual tools is in practice.

Overlap between STQP and STCA

There seems to be overlap or at least a strong interdependence between parts of the STQP and the STCA. The STCA seems to be prerequisite to complete the sections of the STQP that are concerned with the tool qualification methods and the required Tool Confidence Level. On the other hand, the documentation of the tool use cases and the means for detecting malfunctions or erroneous output seem to be input for the tool classification. In the final standard, the authors would like to see a clearer separation of concerns between the two documents and a suggested order in which they should be created.

No established tool qualification best practices

The reported ISO 26262 tool qualification activities were carried out at a time where no reference qualifications were available. The authors believe, that the definition of suitable verification and validation measures to be used in combination with a qualified tool is a means to provide practitioners with the necessary guidance to successfully utilize these tools in projects that need to comply with the requirements of ISO 26262. Until common best practices have been established, the chosen qualification approach could be used as a reference for other tool qualifications.

4 Summary and Conclusions

With the advent of ISO 26262 automotive practitioners need to figure out how implement the tool qualification requirements of this standard in practice.

The authors reported on their experiences with one of the first (if not the first) ISO/DIS 26262 tool qualifications of commercially available production code generation and verification tools.

The successful ISO/DIS 26262-8 tool qualifications of MathWorks Real-Time Workshop Embedded Coder, PolySpace Client for C/C++ and PolySpace Server for C/C++ demonstrated the feasibility of applying this functional safety standard to both development and verification tools.

The author's believe that the definition of suitable verification and validation measures to be used in combination with the qualified tools provides practitioners with the necessary guidance to successfully apply Model-Based Design and advanced code verification tools in projects that need to comply with the requirements of ISO/DIS 26262.

Appendix: ISO/DIS 26262 Tool Qualification and Work Products

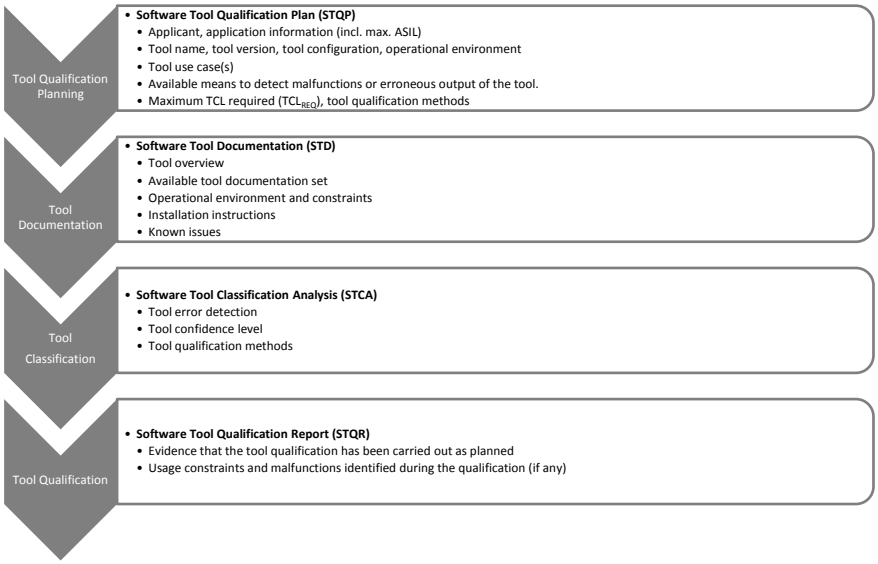

Fig. 3: ISO/DIS 26262 Tool Qualification and Work Products (Overview)

References

[BB09] M. Beine, A. Bärwald: Einsatz zertifizierter Codegenerierungswerkzeuge in sicherheitsgerichteteten Entwicklungen. Safetronic 2009, Munich, Germany, 2009.

[Con07] M. Conrad: Using Simulink® and Real-Time Workshop® Embedded Coder for IEC 61508 Applications. White Paper, Safety Users Group, 2007 www.safetyusersgroup.com/documents/AR070002/EN/AR070002.pdf

[Con09] M. Conrad: Testing-based translation validation of generated code in the context of IEC 61508. Formal Methods in System Design, 2009. DOI 10.1007/s10703-009-0082-0

[CS09] M. Conrad, G. Sandmann: A Verification and Validation Workflow for IEC 61508 Applications. SAE Techn. Paper #2009-01-0271, SAE World Congress 2009

[DO-178B] RTCA/DO-178B. Software Considerations in Airborne Systems and Equipment Certification. 1992

[Glö08] T. Glötzner: IEC 61508 Certification of a Code Generator, ICSS2008

[HB07] V. Hilderman, T. Baghai: Avionics Certification – A complete guide to DO-178 (Software) DO-254 (Hardware). Avionics Communications Inc., VA, USA, 2007

[IEC 61508-3] IEC 61508-3:1998. Int. Standard Functional safety of electrical/ electronic/ programmable electronic safety-related systems - Part 3: Software requirements. 1998.

[IEC 61508-6] IEC 61508-6:1998. Int. Standard Functional safety of electrical/ electronic/ programmable electronic safety-related systems - Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3. 1998.

[ISO/DIS 26262-8] ISO/DIS 26262-8:2009. Draft International Standard Road vehicles — Functional safety - Part 8: Supporting processes. 2009.

[JG03] F. Junker, G. Glöe: Guaranteed Product Safety According to the IEC 61508 Standard. RealTimes 1/2003

[KM03] M. Kühl, K. D. Müller-Glaser: Qualitätssicherung und Zertifizierung beim Softwareentwurf sicherheitskritischer Kfz-Steuergeräte mit X-By-Wire-Technologie (Quality Assurance and Software Certification in respect to Software Construction of Safety Critical X-by-Wire Systems). Elektronik im Kraftfahrzeug, Baden-Baden, Germany 2003. VDI Berichte 1789/2003:467-475

[MBD] Model-Based Design web page. The MathWorks Inc., www.mathworks.com/applications/controldesign/description

[MISRA-C] MISRA-C:2004. Guidelines for the use of the C language in critical systems. 2004.

[KZ09] A. Kornecki, J. Zalewski: Certification of software for real-time safety-critical systems: state of the art. Innovations Syst Softw Eng (2009) 5:149–161

[PM99] Y. Papadopoulos, J. A. McDermid: The Potential for a Generic Approach to Certification of Safety-Critical Systems in the Transportation Sector. Journal of Reliability Engineering and System Safety 63 (1999) 47-66

[RTW-EC] Real-Time Workshop® Embedded Coder™ product page. The MathWorks Inc., www.mathworks.com/products/rtwembedded

[Sau09] J. Sauler: Die ISO 26262 für Automotive kommt! Elektronikpraxis TV (in German), 2009
Part 1: www.youtube.com/watch?v=wqbNrgRcEVo
Part 2: www.youtube.com/watch?v=vWkdIRINb8o

[SLM09] S. Schneider, T. Lovric, P. S. Mai: The Validation Suite Approach to Safety Qualification of Tools. SAE World Congress 2009, Detroit, MI, USA, 2009.

[TMW08] The MathWorks Real-Time Workshop Embedded Coder Certified By TÜV SÜD Automotive GmbH. Press Release, The MathWorks, Inc., 2008
www.mathworks.com/company/pressroom/articles/article31189.html

[TMW09] The MathWorks Real-Time Workshop Embedded Coder and PolySpace Products Qualified According To ISO 26262. Press Release, The MathWorks, Inc., 2009
www.mathworks.com/company/pressroom/articles/article39270.html