Solutions for Embedded Systems Development

Qualification of a Software Development Tool According to ISO26262

Tool Qualification for the New Automotive Standard from a Tool Manufacturer's Perspective

Erol Simsek, iSYSTEM

Summary

Chapter 8-11 of the ISO26262 standard, a version of IEC61508 specifically "adapted" to automotive, is dedicated to confidence in the use of software tools. This standard has a specific impact on not only vehicle manufacturers and their suppliers but also on the development processes of tool manufacturers and thus the development process of the actual software tool.

This article depicts the organizational and technical measures a manufacturer like iSYSTEM has derived from the current standard to support automotive software development departments in the preparatory stage and especially during tool qualification. Experience from earlier tool qualifications in aviation (DO-178B) have been incorporated in this process.

The measures that are discussed in this paper have a positive impact on the development process of a software tool manufacturer to the effect that processes are reassessed and optimized. The outward transparency of these processes is thus enhanced, which is not only for the benefit of the automotive industry but for the benefit of all customers.

Overview

ISO26262 is a standard that shall provide for functional safety in vehicles. Parts of this standard describe measures to be taken by automotive suppliers and/or manufacturers in order to minimize the risk of the software tools used introducing or failing to detect errors in an end product. To this end, extensive consideration and risk assessment regarding the tools used in the development and test process are already required in preparation of a development. The automotive supplier/manufacturer can be partly relieved of these considerations if the tool manufacturer takes over of some preliminary tasks. Independent organizations moreover offer certificates to prove that a software tool is suitable for use in the development of a safety-related application. To put it in a nutshell: it's all about liability.

ISO26262 Recommendations Regarding the Use of Software Tools

With the term software tools, the standard refers to software development tools and software verification tools.

Software development tools are tools that are used to develop software and can introduce an error directly in the end product (code generators, compilers, software libraries, requirements management tools, modeling tools, …).

Software verification tools are tools that are used to test and verify software and thus do not introduce an error directly in the end product, however, could prevent those errors from being detected (simulators, emulators, test tools for unit test, code coverage analysis, …).

According to ISO26262 part 8-11 "Confidence in the use of software tools", it is necessary to asses resp. examine, in the course of a safety-related project, if a malfunction of the software tool to be used can introduce or fail to detect errors in the end product. This is referred to as the so-called tool impact (TI).

	TD1	**TD2**	**TD3**
TI1	TCL1	TCL1	TCL1
TI2	TCL1	TCL2	TCL3

Image 1: Determining the TCL (tool confidence level) – source: ISO26262-8-11

For risk assessment, there are two defined tool impact states (TI): TI1, no risk, and TI2, risk. The "tool error detection level" (TD) is identified in the next step. Confidence is ranked according to methods and measures that shall detect a tool's malfunction defined according to the standard ("introduce or fail to detect errors", ISO26262-8-11). This can be pre- or postverification steps or redundancy or diversification techniques (e.g. using a tool with similar functionality) in the process. TD1 – TD3 define the different degrees of confidence, with 1 being the highest and 3 the lowest degree. Using a specific table in part 8 (image 1), the user can subsequently determine the so-called tool confidence level (TCL) and, if required, derive suitable tool qualification methods. TCL2 and TCL3 require qualification of a tool (corresponding to the automotive safety integrity level determined, ASIL A-D). The standard recommends the following methods:

- Increased confidence from use
- Evaluation of the tool development/test process of a software tool
- Functional validation of the software tool
- Development in accordance with a safety standard

TCL2		A	B	C	D
1a	Increased confidence from use	++	++	++	+
1b	Evaluation of the tool development/test process of a software tool	++	++	++	+
1c	Functional validation of the software tool	+	+	+	++
1d	Development in accordance with a safety standard	+	+	+	++

TCL3		A	B	C	D
1a	Increased confidence from use	++	++	+	+
1b	Evaluation of the tool development/test process of a software tool	++	++	+	+
1c	Functional validation of the software tool	+	+	++	++
1d	Development in accordance with a safety standard	+	+	++	++

Image 2: Possible tool qualification methods according to the respective ASIL level – free translation from the source ISO26262-8-11, + recommended, ++ highly recommended

Assessment of Tool Qualification Methods

From the perspective of a manufacturer of debug tools with integrated software test functions, the following can be observed regarding the methods given in image 2:

Increased confidence from use

The software tool is normally used in a specific version and configuration, for a dedicated microcontroller or microprocessor and in combination with other software tools (e.g. compilers). Only in the rarest cases can the very same tool setup be used for new projects, which is partly due to the multitude of microcontrollers/microprocessors - some of which are even brand-new. For this reason, software tool manufacturers often pursue a dynamic release policy. Typically, software versions of a tool that were partially tested in the field are used in addition to an officially and thoroughly tested software release. Not to mention potential hotfixes - pragmatically introduced changes to the software functionality, e.g. to meet customer requests. This is absolutely legitimate and “state of the art”. Such software can be used safely if suitable documentation is available. However, the argument of increased confidence from use is slightly weakened in this context.

Nonetheless, increased confidence from use can be examined in the scope of tool qualification – provided that similar processes were successfully implemented before and that a software tool has been extensively used worldwide within a company for an extended period of time, etc. In addition, cooperation between customer and supplier has to be assessed in the context of increased confidence from use. Mutual transparency, openness and a healthy degree of pragmatism often form a sound basis of such an assessment. From a legal perspective, however, this does not help any further, and this is probably the crucial factor. For this reason, image 2 lists more than 2 methods that must be considered for a tool qualification process; also in combination.

Evaluation of the tool development/test process of a software tool

Most software tool manufacturers have defined their processes and implement them actively. However, only a few of these companies have probably implemented a process model according to e.g. SPICE or CMMI. More companies are ISO9001:2008 certified. Processes can basically be assessed by the customer, either by a simple review of the quality handbook and suitable process descriptions or even through external audits at the manufacturer. The related effort is an issue here as well, however, this procedure is a good basis for building real confidence. Would it help if an independent third party took over these tasks? Basically yes, provided that the respective processes are established for audit (in accordance with ISO9001) or even assessment (in accordance with a process model like SPICE). The related safety aspects of the standard can be considered additionally. However, it might turn out to be more efficient for the tool manufacturer to conduct the discussion themselves. Non-liability is definitely not achieved with certificates only.

Functional validation of the software tool

This is the time-consuming and probably cost-intensive part. As a countermeasure, the software tool manufacturer can take over specific preliminary tasks. The solution presented in the following is the methodology of iSYSTEM AG and particularly refers to debug tools with integrated test functionality. The methodology can basically be transferred to other software tools and is thus generic.

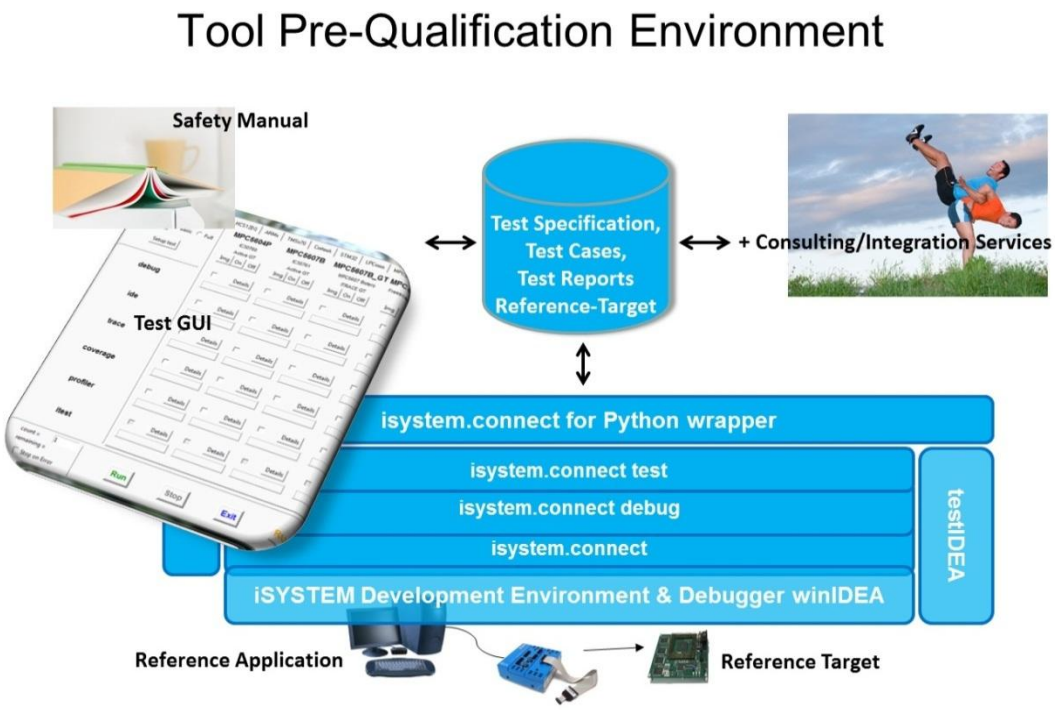

Image 3: iSYSTEM pre-qualification environment for on-chip debug and trace tools

If the function of an iSYSTEM tool has to be validated based on ISO26262, iSYSTEM provides a "tool pre-qualification" environment. It comprises reference hardware for the different microcontrollers and related test cases to check the following functions of an on-chip debug and trace tool:

- standard debugging and IDE functions (memory read, write, step, memory dump, download, flash programming, etc.)
- advanced debugging with trace and profiling (especially time measurements)
- software test with code coverage and unit test

It is basically an API based (application programming interface) solution as shown in image 3. The iSYSTEM hardware tools are user controlled through an integrated development environment, e.g. for software development or for the software test of an application. To implement a remote control function for test automation, iSYSTEM provides the generic interface isystem.connect with different abstraction levels (including control via script languages). Based on this structure, iSYSTEM has established its own regression test environment. The test GUI and various test cases are written in Python.

Reference target systems and the related reference applications are used in order to derive suitable reference test cases based on the exactly known behavior of the hardware and the respective reference application. These reference test cases are also adapted to a wide variety of compiler combinations. The expected values obtained in the test can thus be predicted accurately.

The iSYSTEM test automation solution can now be efficiently tailored to the requirements of a tool qualification process in a safety-related project resp. to a customer's target system. The customer can do this by himself or in cooperation with iSYSTEM.

As an "organizational measure", iSYSTEM provides suitable documentation that creates transparency regarding the development and test process within iSYSTEM. It is basically a "safety handbook" for the use of iSYSTEM tools, created individually with the customer, and consists of:

- descriptions of features /functions used in a safety-related project
- explanations of the usage of these functions
- manuals for software and hardware use, "Getting Started" guides", etc.
- descriptions of system requirements, generic and in respect to a target processor
- descriptions of known workarounds for a specific microprocessor

The safety handbook provides the developer with information on how to use an iSYSTEM tool for the development of a safety-critical application.

Development in accordance with a safety standard

For the time being, this method is probably not implemented by many software tool manufacturers. The relevant safety standards should be examined such that those parts of the standard can be determined that can be appropriately incorporated in the development and test process of a software tool manufacturer. This again might be best accomplished by discussing possible implementation scenarios with the software tool manufacturer.

Outlook

The new automotive standard ISO26262 introduces an alteration to tool qualification that is interesting for all parties - the risk of using software tools in respect to specific requirements has to be considered in preparation of a development. Software tool manufacturers are not excluded either, i.e. these manufacturers are expected to implement suitable improvements and extensions in their own development and test processes.

http://de.wikipedia.org/wiki/ISO_26262
http://de.wikipedia.org/wiki/IEC_61508
http://de.wikipedia.org/wiki/Spice_(Norm)
http://de.wikipedia.org/wiki/Capability_Maturity_Model_Integration