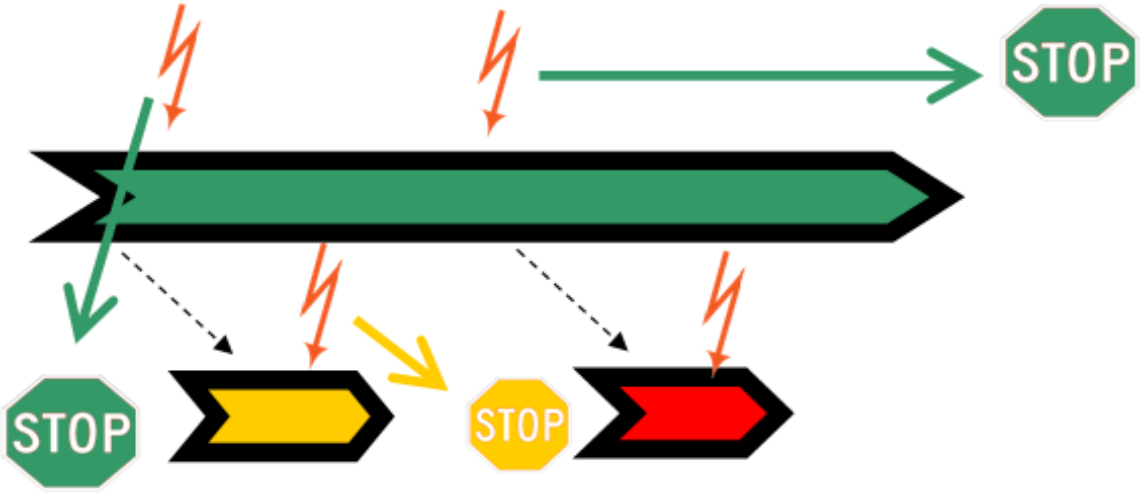
User Manual: Qualification Kit for Testwell CTC++



Version:	1.2
Date:	2014-05-28
Status:	In Progress / reviewed / presented / **final**
Author:	Dr. Oscar Slotosch
File:	KitUserGuide.docx
Size:	57 Pages

Verifysoft TECHNOLOGY

History:

Version	Date	Status	Autor	Change
0.1	2012/02/09	In Progress	Slotosch	Template created
0.2	2013/10/16	In Progress	Slotosch	Updated to new QST
0.3	2013/11/06	Reviewed	Slotosch	Integrated feedback from M. Wildmoser
0.9	2013-11-19	Presented	Slotosch	Adapted to Testwell CTC++
0.9.1	2013-12-03	Presented	Slotosch	Updated Images and some texts
1.0	2013-12-21	Finalized	Slotosch	Integrated Feedback from Verifysoft
1.1	2014-02-16	In Progress	Slotosch	Qualification Extensions
1.2	2014-05-13	In Progress	Slotosch	Updated to new QST version

Contents

List of Figures

1 Scope of this Document

This document describes how to use the qualification kit for the Testwell CTC++, i.e. to qualify the Testwell CTC++ according to ISO 26262, IEC 61508, EN 50128, DO-178C/DO-278A/DO-330 (TQL-5 to ensure that it does not influence the safety of the developed products during its operation negatively.

Since this qualification kit can be adapted to the process of the user and since it can be extended there are other related documents (see the documentation plan in Section 4).

This document contains an overview of the qualification kit and explains how to use it and how to create the required documents to demonstrate the qualification of the Testwell CTC++ within your safety case.

The safety of tools is achieved within three steps

1. Tool evaluation and possibly qualification,
2. Proper tool installation and
3. Proper tool operation.

The qualification of the tool is achieved within the following steps:

1. Determination of the qualification need (from the tool analysis)
2. Creation of a tool qualification plan (in our case we use validation by test as main qualification method)
3. Execution of the tool qualification according to the test plan
4. Documentation of the tool qualification results in the tool qualification report and the tool safety manual.

Therefore this document is structured as follows

- Method: Model-based tool qualification, see Section 3,
- Documentation Structure, see Section 4,
- Application of the qualification kit, see Section 5,
- Extension of the qualification kit, see Section 6 and

Furthermore the document demonstrates the standard compliance of the kit by tracing against the requirements from the relevant safety standards (see appendix in Section 7).

2 Glossary

This section defines technical terms used within this document.

Term	Definition
Check	possibility to detect an error
Error	in this document used as "potential error"
Error (model) element	representation of an (potential) error in the model
Feature (model) element	representation of a function in the model.
Function	an elementary or composed function of the tool, that can be required in one or more use-cases, e.g. load, save, "perform" functions
Qualification environment	TAU and tests, a validation suite according to ISO 26262
Restriction	possibility to avoid an error
Safety Guideline	Guideline to mitigate some potential errors of the tool. Modeled as a *Check* or *Restriction*, either in an usual *UseCase* or *Feature* of the *Tool*, or in a separate, virtual *Feature* that can be required (added) by any use case of the same tool. Safety Guidelines are listed in the tool classification report.
software off-line support tool (IEC 6108)	According to IEC61508-4-3.2.11: software tool that supports a phase of the software development lifecycle and that cannot directly influence the safety-related system during its run time.
TAU	Test Automation Unit: executes tests for the test suite
TD	Tool Error Detection (TD) probability for a potential error to be detected / avoided in a defined process TD1=high detection probability, TD2=medium detection probability, TD3=low or unknown detection probability
TCL (ISO 26262-8)	Tool Confidence Level (ISO 26262): required confidence in the tool when used in the analyzed tool chain TCL1=low confidence required , TCL2=medium confidence required, TCL3=high confidence required[1]
Test	Single test with result PASS/FAIL/ABORT
Test Directory	A directory containing one or more test (directories)
Test (model) element	Representation of a test directory in the model including a test description that specifies it
Test Suite	structured set of single tests
Test Plan	list of test (directories) to be executed
Tool	a development tool according to ISO 26262
Tool Chain	a collection of tools, not necessarily forming an input/output chain

[1] Of course once the tool with TCL>1 have been qualified, the TCL can be regarded as existing tool confidence for the qualified ASIL rather than required tool confidence.

Tool classes (IEC 61508-4)	Software off-line support tools are classified into the following tool classes: T1: generates no outputs which can directly or indirectly contribute to the executable code (including data) of the safety related system T2: supports the test or verification of the design or executable code, where errors in the tool can fail to reveal defects but cannot directly create errors in the executable software T3: generates outputs which can directly or indirectly contribute to the executable code of the safety related system.
Tool Classification	determination of the required tool confidence level (ISO26262: TCL or IEC 61508: tool classes)
Tool Evaluation	or tool criteria evaluation: see tool classification
Use-Case	the purpose of using the tool in development process
Use Case (model) element	representation of an use-case in the model
Virtual Feature	A *Feature* is called virtual, if it's virtual attribute is set to true. *Virtual Feature*s are modeled in a *Tool*, but are not implemented in the tool. They are used to model safety guidelines (documents) and can be added flexible as required features to use cases to denote that the use cases follow them. Virtual feature do not have errors.

Note that elements, relations and actions from the model that have a formal semantic in the TCA are written in capital and with italic font, e.g. "*Error* element", or "*Export -> Excel Review*".

3 Method: Model-Based Tool Qualification

Tool qualification shall demonstrate that the tool is qualified to fulfill its task safely. Usually this is achieved by fixing a reference process and a small set of reference use cases and demonstrating by tests that the tool is working correctly.
The problem of this approach is that in tool chains nowadays the tools are used in many different ways, that a fixed reference process or use case cannot be considered if several qualifiable tools shall be integrated into one tool chain. Therefore the tool qualification kit for the Testwell CTC++ is based on a model that can be adapted by the user in a flexible way such that the tool can also be qualified within a user-defined setting.
The model-based tool qualification approach offers the following benefits:

- Formalization: The model is precise and decisions e.g. on assumptions, can be stored. Furthermore it allows the user to express complex situations like alternative mitigations or variants in the tool chain in a clear way.
- Flexibility: The user can configure his use-cases by selecting tool features and applicable mitigations. He has to run only the required qualification tests.
- Reusability: The tool model (including the error mitigations) can be reused in the use case definition. Furthermore the tool model can be combined with other tools to reduce the qualification need.
- Consistency: The model can be checked for consistency to avoid wrong assignments from mitigations to errors or to detect missing descriptions or deviations from the general error model.
- Automation: The modeling tool can determine the confidence required in the different use cases, it can help in the management of models (e.g. using Excel interfaces, merging of models, …), generation of reports, review checklists and test plans.
- Analyzability: The model can be used to analyze different situations with different variants of the tool chain, e.g. by adding a new tool, or removing a check. Furthermore the tool can analyze the costs of mitigations to find optimal solutions.

Therefore the model-based tool qualification kit is based on the model.
There are three processes involved in tool qualification:

1) Classification of the tool (within the process) and determination of the confidence needs
2) Tool Qualification, in this case by application of the qualification kit that validates the tool
3) Building or extending the qualification kit

This qualification kit supports all three processes. Classification and qualification are done by the qualification support tool that is contained in this qualification kit, while extension can be done with the Tool Chain Analyzer (TCA) tool. The TCA tool can be freely downloaded from www.validas.de/TCA.html for Windows, Linux and MacOS, such that every user can use the kit easily.

4 Documentation Structure

The safety standards (ISO 26262, IEC 61508, DO-178/DO-330) require the user to analyze the tools used for the development of safety-critical products. The result of the analysis is a requirement on the reliability of the tool stated in the tool criteria evaluation report.
The confidence is determined by an analysis of the use cases of the tool as used within the development process. If the tool has an impact on the safety of the product, all potential errors within the used features are analyzed for how they can be detected or avoided within the process. If there is no high probability for detecting or avoiding the errors, the tool has to be qualified to ensure the absence of these errors.

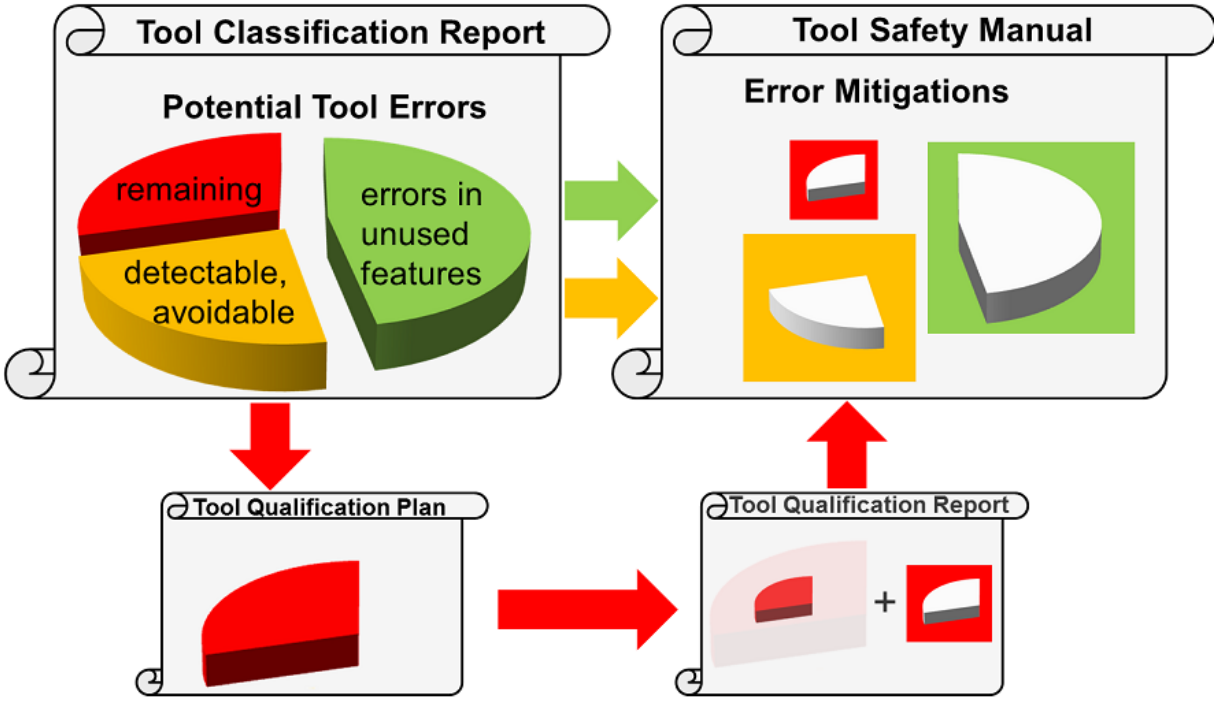


Figure 1: Derivation of Tool Safety Manual Contents

The tool safety manual for a tool has to contain the mitigations against all potential tool errors that are considered during tool evaluation [TCR]. The errors can be grouped into the three classes (see Figure 1):

- Potential errors in unused features (green in Figure 1)[2]: Using these features is prohibited in the tool safety manual.
- Potential errors with mitigations: detections and restrictions (yellow in Figure 1): These mechanisms are described in the tool safety

[2] Note that the analysis of potential errors in unused functions is not required, but the features need to be identified.

manual, especially if the checks/restrictions have to be triggered by the user of that tool.
- Remaining potential errors (red in Figure 1): Demonstrating their absence has to be the goal of the tool qualification (tool qualification plan). The tool qualification report possibly shows some concrete errors that are instances of the potential error classes. The qualification report contains proposed workarounds for these concrete errors that have to be part of the safety manual (together with the workaround for other already known relevant errors).

The tool safety manual therefore has to contain the following information:
- Allowed features and configurations of the tool
- For potential errors that might occur in required features and that are not excluded by tool qualification: Requirements to apply checks and restrictions to mitigate potential tool errors
- Workarounds for known errors and errors found during qualification
- Other information required by the standards to identify the tool exactly (version, configuration, etc.).

The tool qualification plan has to ensure that the identified potential errors of Testwell CTC++ that are not detectable / avoidable cannot occur. This is done by applying a validation suite in a systematic way that shows the absence of these potential errors.

Since the tool Testwell CTC++ shall be qualified using validation accruing to this qualification plan we have to provide the following documents:
- *Test Plan*: to plan the execution of tests
- *Test Report*: contains the test results
- Test Automation Unit Manual: To execute the planned tests cases correctly
- *Test suite validation and verification documents (plan and report)*: to ensure that the test suite shows the absence of the potential errors if passed successfully

The documents that depend on the model are typed using *cursive font*.

In the case that the model and the validation suite needs to be extended and new test cases need to be produced and validated, the following documents are required, or need to be extended:
- *Test specifications* including a test strategy to show the absence of the absence of the potential errors.
- *Test suite V&V plan & report*

The test specification is part of the model (descriptions). The test suite needs validation against the potential errors of the model and verification against the implementation using a review. This quality process creates the confidence into the effectiveness of the test suite. The V&V documents for the test suite are contained in the qualification kit to demonstrate the

confidence to the user. If the test suite is extended these documents shall also be extended.

Figure 2 shows the relation between the documents and their variability, i.e. which are constant and which depend on the use case: It describes how to derive the safety manual by a validation suite that consist of tests that show the absence of the identified critical errors in the tool evaluation report. Depending on the used features of the tool and the applied mitigation measures this set of errors might vary. For every required test (or group of tests) that show the absence of one or more errors there needs to be a test specification (including a test strategy) that explains how the absence of the errors is ensured if the tests pass. The tests in the test suite need to be validated to conform to the test specification. This is planned in a V&V plan of the kit and documented in the V&V report. Having a V&V report is the prerequisite for applying the validation suite to a use case. In Figure 2 the use case specific documents are in a green/inner, dashed box where the contents of the qualification kit are in the outer/blue box. Of course the sequence of creating the documents (indicated by the sequence numbers) starts with the non use-case specific documents in the qualification kit. The tool qualification is planned in the qualification plan and requires executing tests (planned in the test plan) using the test automation unit manual. The test results are documented in a test report which is than analyzed and documented in the qualification report.

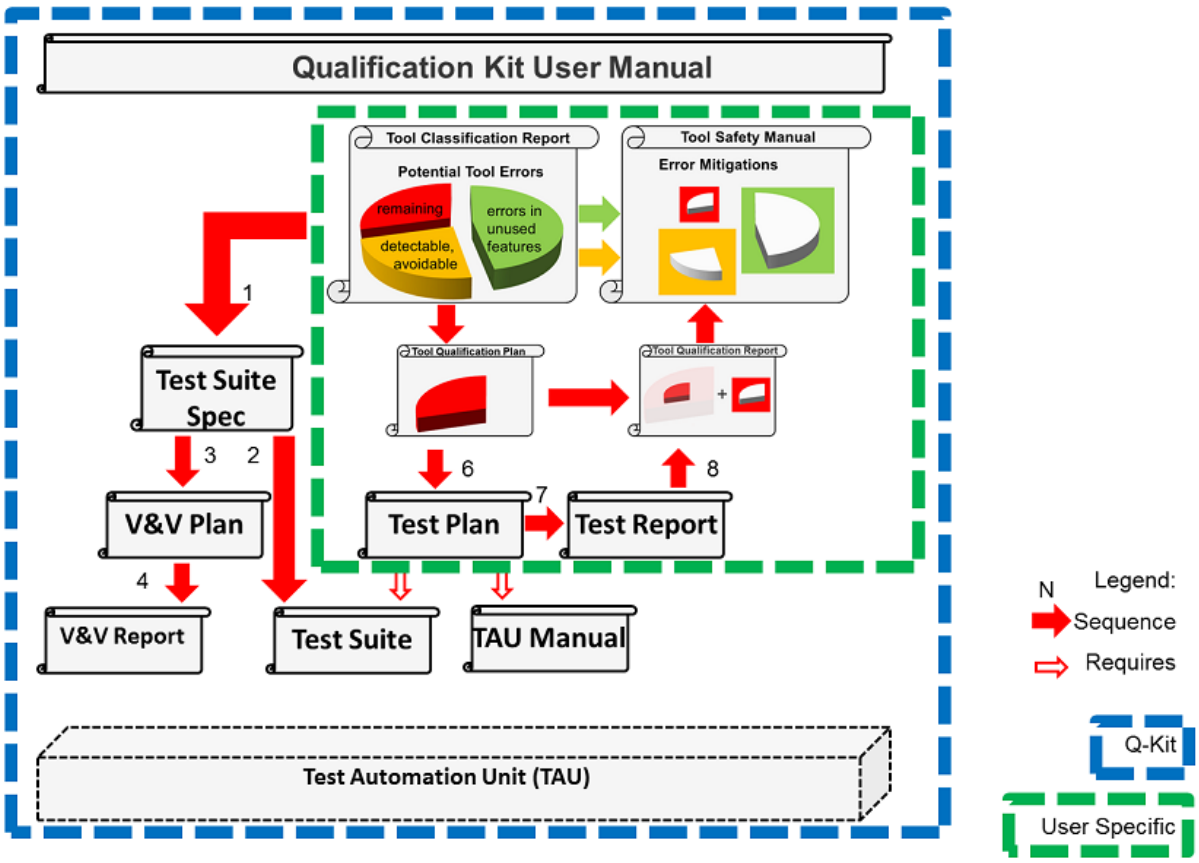


Figure 2: Documentation Plan

There are many documents in Figure 2 that are required and that need to be adapted depending on the user's process captured in the qualification model by selecting the required tool features and the executed mitigations during the process. The use case specific parts in the user specific documents are generated from the qualification support tool.

5 Application of the Qualification Kit

The tool qualification kit for the Testwell CTC++ is applied using the qualification support tool (QST). This tool determines the qualification need for the Testwell CTC++ and the detection of potential errors. If required the QST prepares the qualification by testing. Furthermore the tool generates the required documentation and artifacts.

This section describes

- the determination of the qualification needs for the Testwell CTC++ and the detection of potential errors (see Section 5.1),
- the qualification of Testwell CTC++, see Section 5.2 and
- the qualification support tool, see Section 5.3

5.1 Determination of Qualification Need and Potential Error Detection

While ISO 26262 requires to analyze the detection of potential errors in the use-cases of the tools, the other standards requires this analysis as part of the tool qualification (see Figure 36 in Section 8). So the analysis is required for any tool that needs qualification, but only in the ISO 26262 it is part of the tool classification. The other standards require to classify the purposes of the tool, this is done using the methods that the tool supports.
The potential errors depend on the features from the tool that are used. The error mitigation probability depends on the applied process, especially which checks and restrictions are applied in the use case of the tool, or within other tools.
The error detection for Testwell CTC++ is determined by selecting the used features of the tool and by selecting the applied mitigations. While the list of features of Testwell CTC++ is constant, the list of mitigation depends on the selected features. For example if no features are selected, than no mitigations are required.
Depending on the availability of test cases and possible mitigations, every feature (strictly speaking also every potential error) has **qualification state**:

- Green: The feature is tested and can be used without usage constraints to mitigate potential errors
- Yellow: The feature cannot be tested and requires usage constraints
- Red: Neither test case nor error mitigations are available. The feature cannot be used without extending the qualification kit.

Whether the tool is qualified successfully depends on

- The commitment to the required mitigations (if required by the selected features)
- The successful execution of the test cases (if required by the selected features)

So the determination of the qualification needs can show either that test cases have to be executed or that usage constraints to mitigate potential tool errors have to be integrated into the processes (or a combination of both situations). The results of both are described in the tool safety manual that describes how the tool can be used safely.

5.2 Qualification

The qualification of the Testwell CTC++ depends on the qualification need. As described in Section 5.1 the qualification depends on the selected features that require either test or mitigation to ensure the absence of potential errors. The result of both activities are integrated to the tool safety manual.
In the case test and mitigations are required the qualification process consists of the following steps:

- Determination of qualification need, see Section 5.1
- Creation of documents (including a test plan)
- Execution of the tests with the TAU according to [TAU_UG]
    - Installation of the TAU
    - Execution of the required tests according the test plan
    - Analysis of test results
- Finalization of documents

The qualification support tool (see Section 5.3) supports the determination of the qualification need and created the documents, such that the user only has to run tests and finalize documents.

5.3 Qualification Support Tool

The qualification support tool is part of the qualification kit. It supports the user during the qualification by the determination of the qualification need and the generation of the qualification documents.

5.3.1 Requirements

The qualification support tool is available for Windows, Linux and MacOS-X systems and requires 1 GB memory. The tool does not require administration right.
For the generation of the images in the documents the graphiz tool has to be installed and the (dot.exe) has to be in the execution path. Graphviz is open source and can be downloaded at http://www.graphviz.org. Graphviz version 2.28 or higher is required.

5.3.2 Installation

The qualification support tool is delivered as zipped file. In the zipped file there is a directory that contains an executable that has to be used to start the qualification tool (see Figure 3).

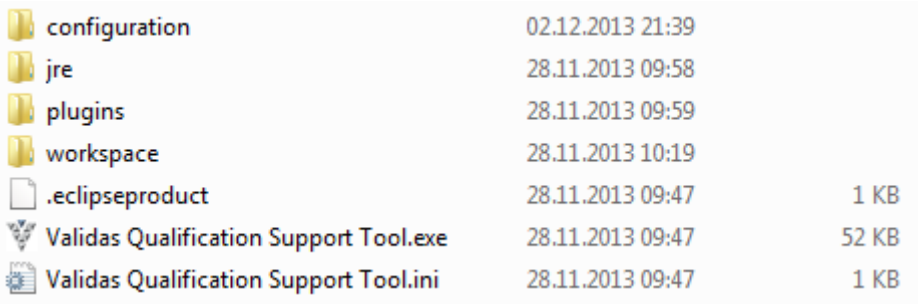


Figure 3: Starting the Qualification Support Tool

5.3.3 Start

The executable has to be double clicked to start the qualification tool. After starting the tool a welcome screen shows as shown in Figure 4.

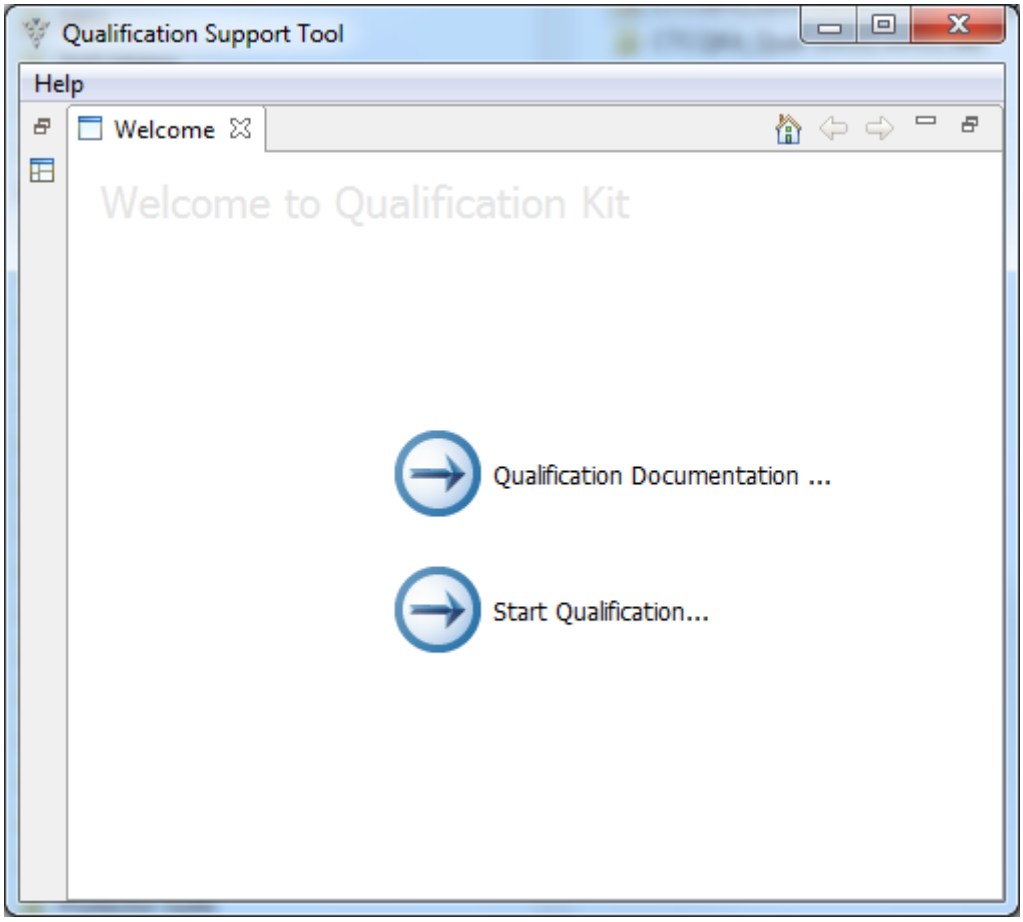


Figure 4: Welcome Screen after Start of Qualification Support Tool

By clicking to the "Start Qualification" button the qualification starts with the choice of a directory into which the qualification shall be executed (see Figure 5).

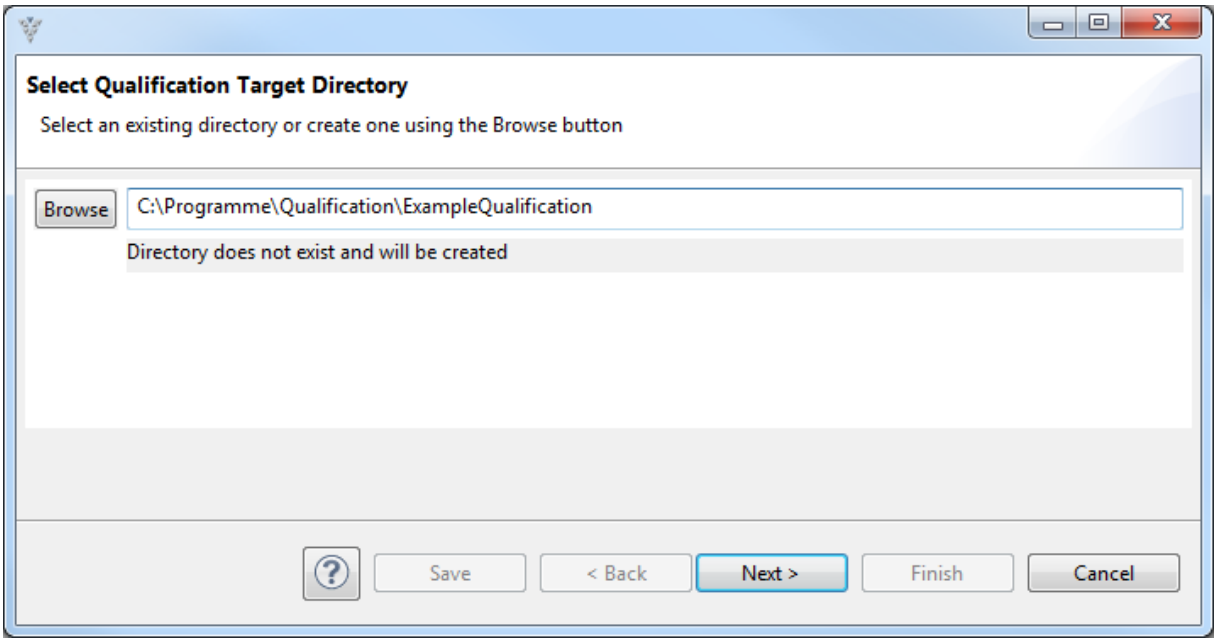


Figure 5: Selection of Qualification Target Directory

The qualification target directory is used to store all qualification artifacts. After the target directory is selected the Qualification Support Tool can assist in the qualification need as described in Section 5.3.4.
If the qualification target does not exist or is empty, a new qualification with the default settings from the QST and the model will be started. If the qualification target is not empty, the qualification will continue with the information (templates, model, status) found in the qualification target. This means that the QST can save the configuration status in the model and continue the qualification later.
If the Help-Button (?) is pressed a context sensitive help is displayed, see Figure 6.

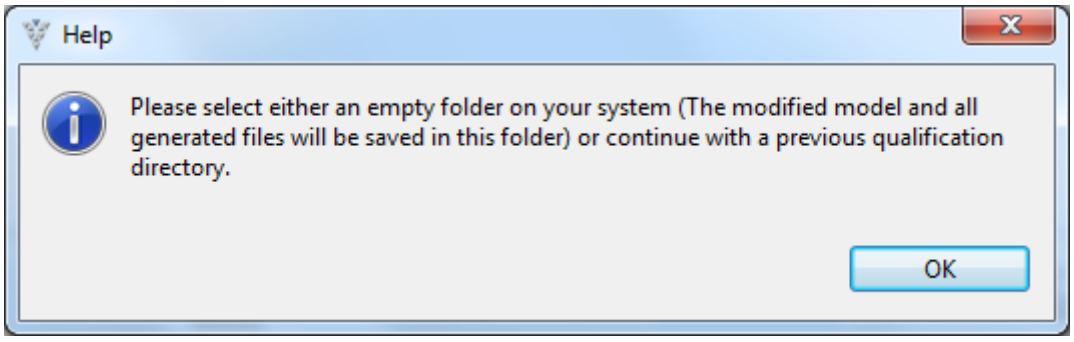


Figure 6: Help Information

5.3.4 Determination of Qualification Need and Error Detection

To determine the qualification need the QST asks for the standard according to which the tool shall be classified & qualified, see Figure 7.

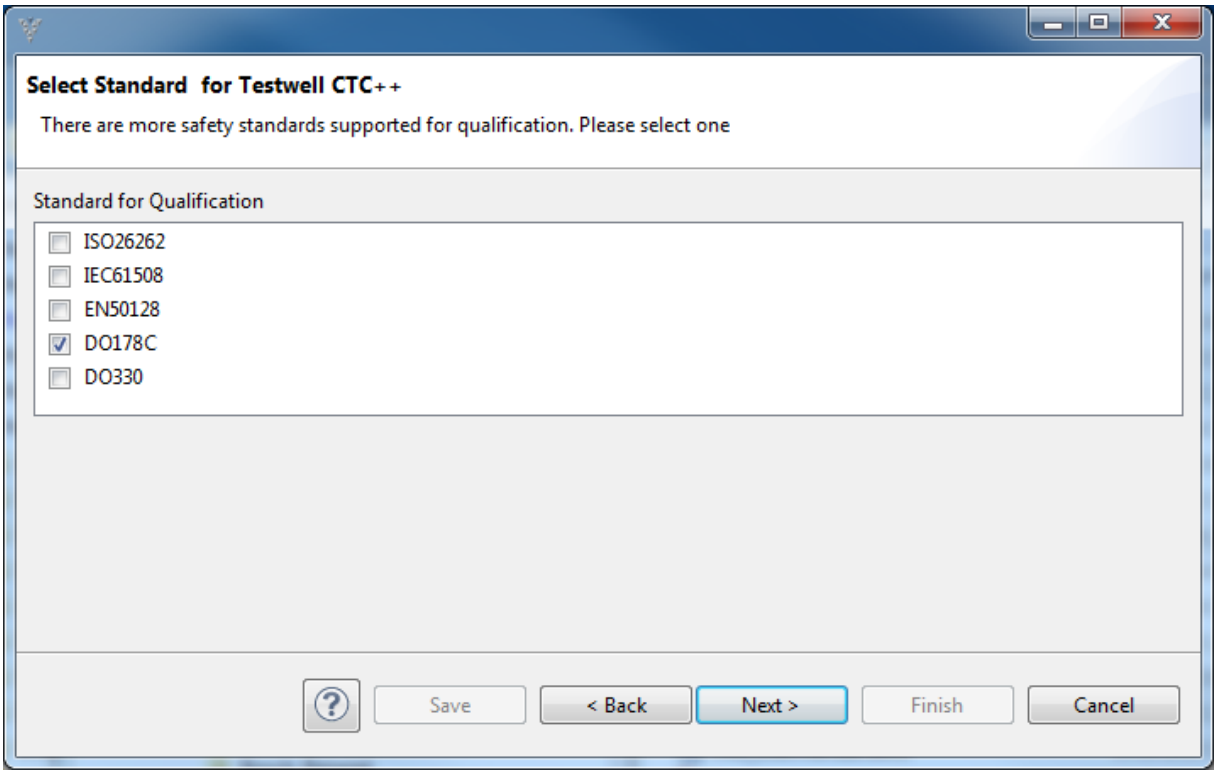


Figure 7: Selection of Safety Standard

The next step is to select the variant of the tool chain. Figure 8 shows the selection dialog of variants that can be in the tool, for example for testing on the PC or testing on the target there are different artifacts and features of the test tool used.

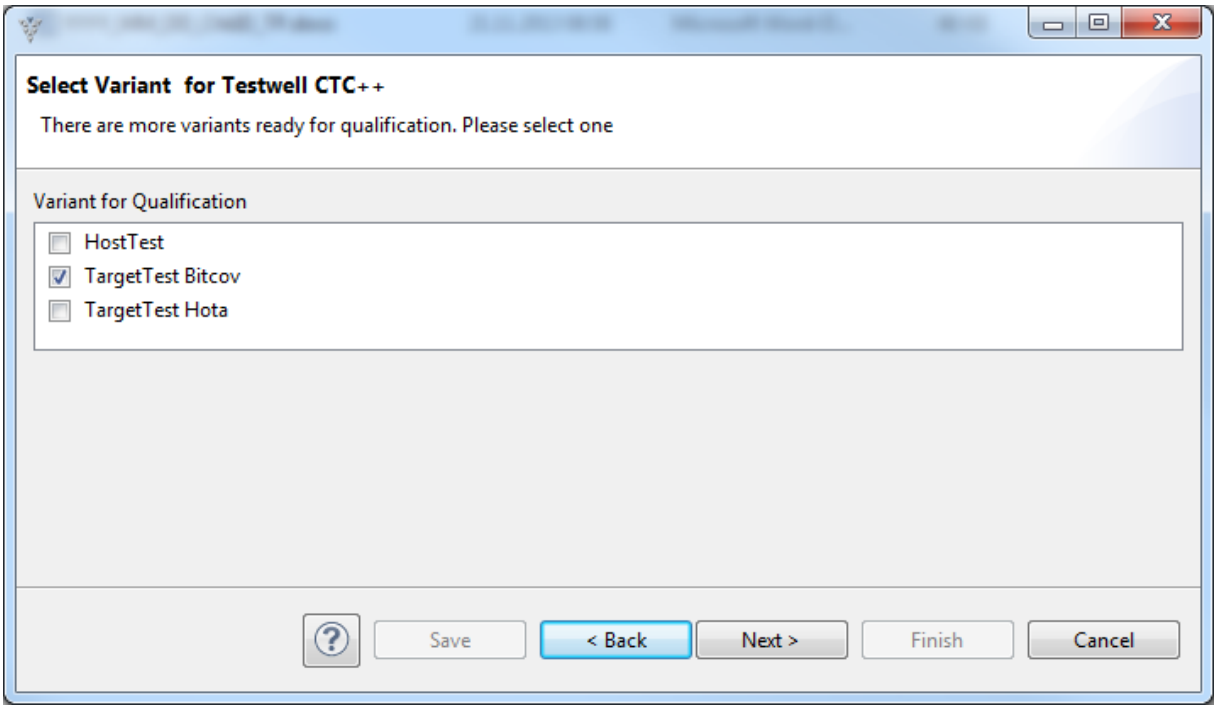


Figure 8: Selection of Variants

Next page is the selection of tools and use cases for qualification. Every use case of the tools will be configured and prepared for qualification. The following identification selection and use-case configuration page will be

repeated for each use case that is defined in the tool and use case selection page, see Figure 9. It shows the list of available tools that can be selected for qualification. Note that if the qualification need can be inferred according to the chosen standard from the model it is depicted in brackets behind the tool name.

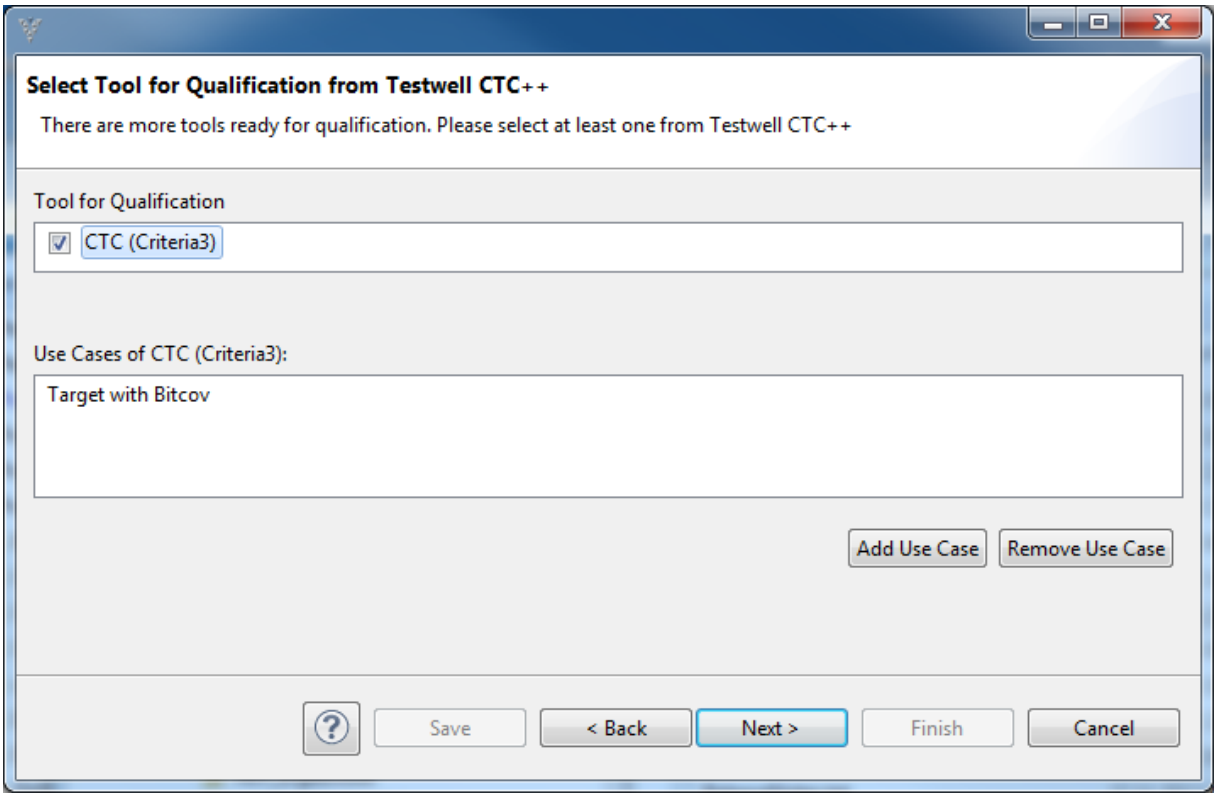


Figure 9: Tool & Use-Case Selection

The selection of the version / identification of the tool is the next step, see Figure 10.

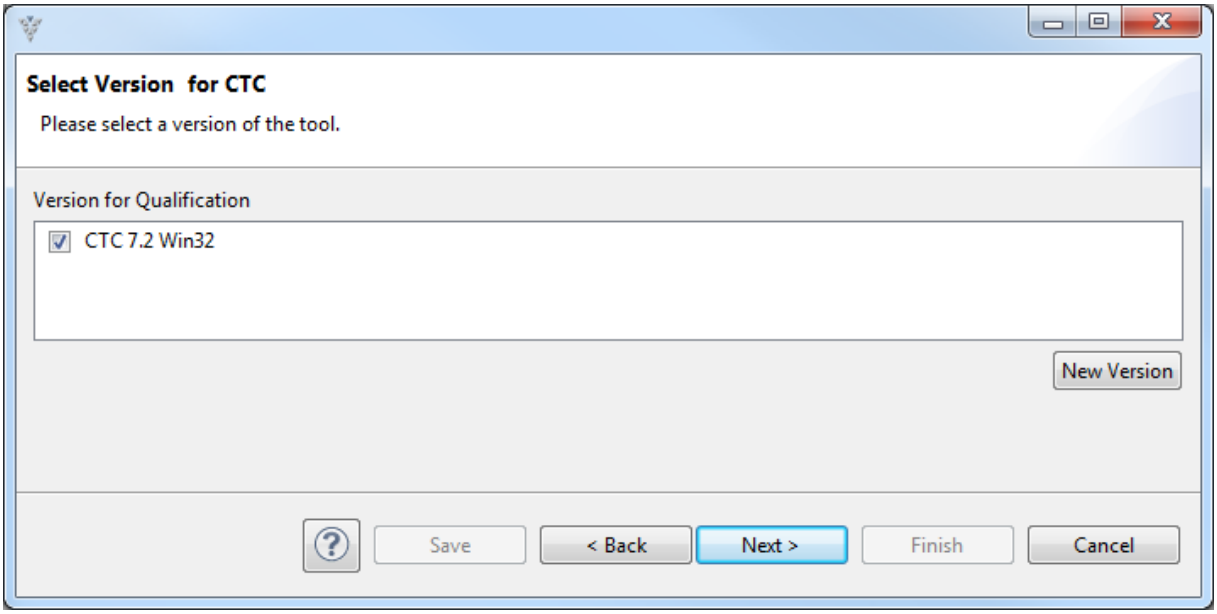


Figure 10: Version Identification Selection

Note: if there is no version information (“Identification”) in the model, the QST will display an empty selection list (see Figure 11) and enable the next button only after new Version information has been entered (see Figure 12). New versions can also be added into non-empty list by pressing the new Version button.

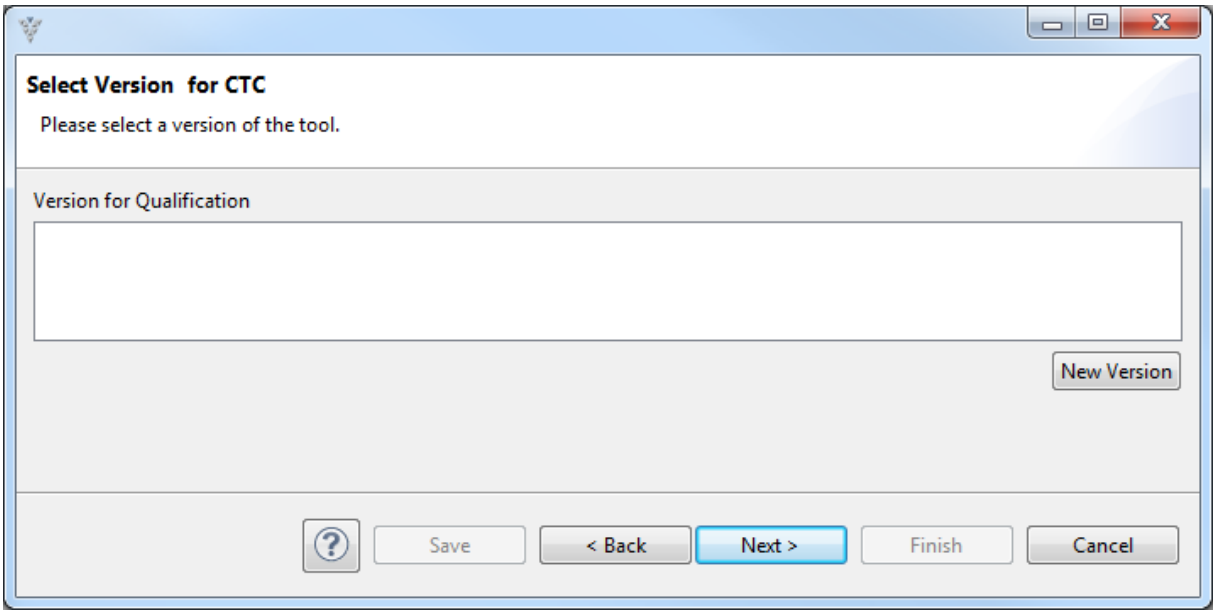


Figure 11: Empty Version List

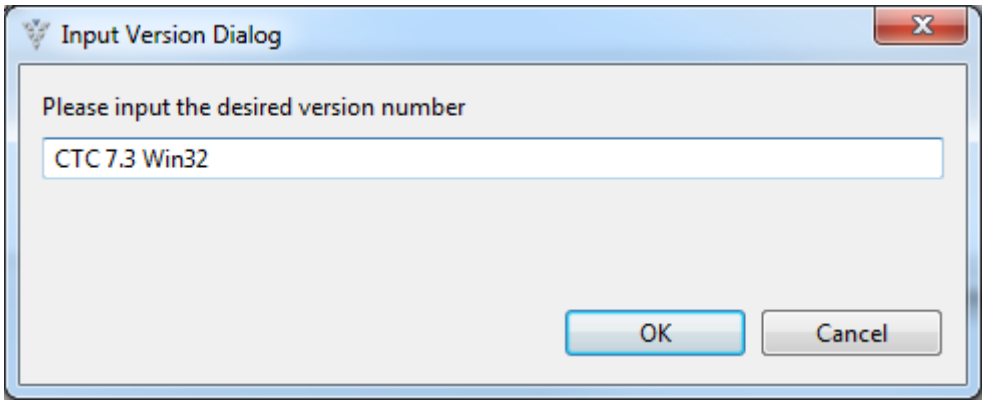


Figure 12: New Version Dialog

After the new version has been entered the user can add a file containing the known bugs into the model (see Figure 13), e.g. the revision history file. The contained information will be added to the model and printed out in the generated tool safety manual.

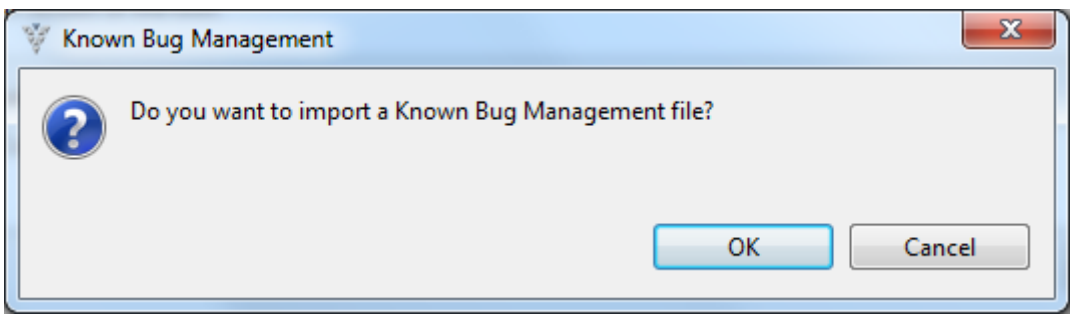


Figure 13 Known Bug Import Dialog

The determination of the qualification need is done in two steps:
1) selection of the features (for all tools with qualification need)
2) selection of mitigations / tests for the potential errors in the selected features (also for all tools with qualification need).

The selection of the features for the tools is done in the feature selection page (see Figure 14). It shows the use cases of the tools (left side) and the available features of the tools (right side). The features have three different colors:
1) green colored features that can be used without constraints, since they are testable
2) pink colored features that can be used with some constraints (mitigations)
3) red colored features (if available)
The preselected features are those from the default model of the tool.
In the bottom there is an information window that shows information.

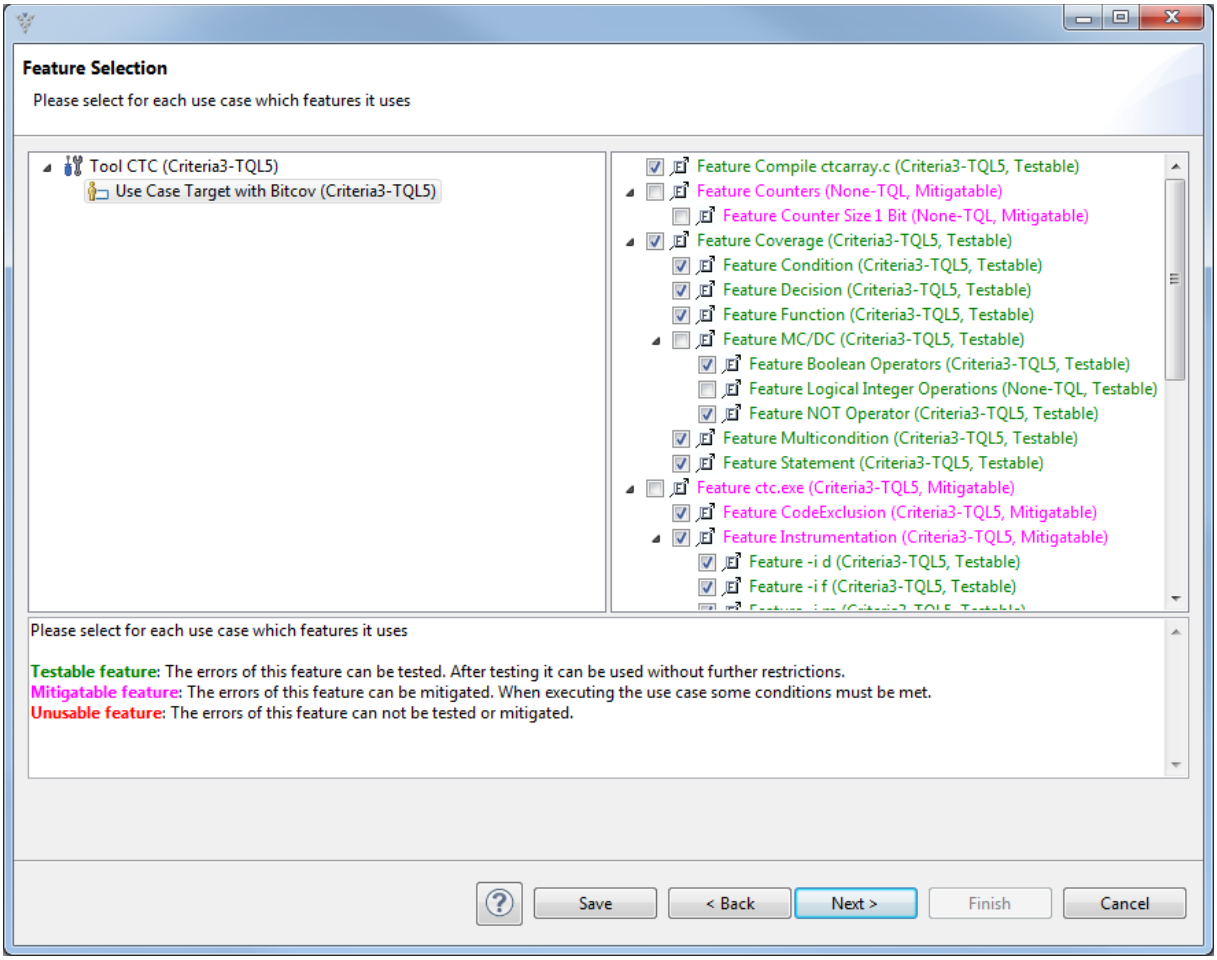


Figure 14: Feature Selection Page

The selection of the mitigations is the next step (see Figure 15). The page contains the list of potential errors on the left side. After selecting an error

the available mitigations are show in the right side and can be selected or deselected by stetting/unsetting their check marks. If errors are mitigated they are marked as mitigated in the left side. In order to facilitate the selection of safety guidelines it is possible to show only the remaining errors (see difference in error list between Figure 15 and Figure 16), i.e. the untestable errors that have no mitigations. This can be done by toggling the button "Only show remaining errors" above the list of errors on the left side.

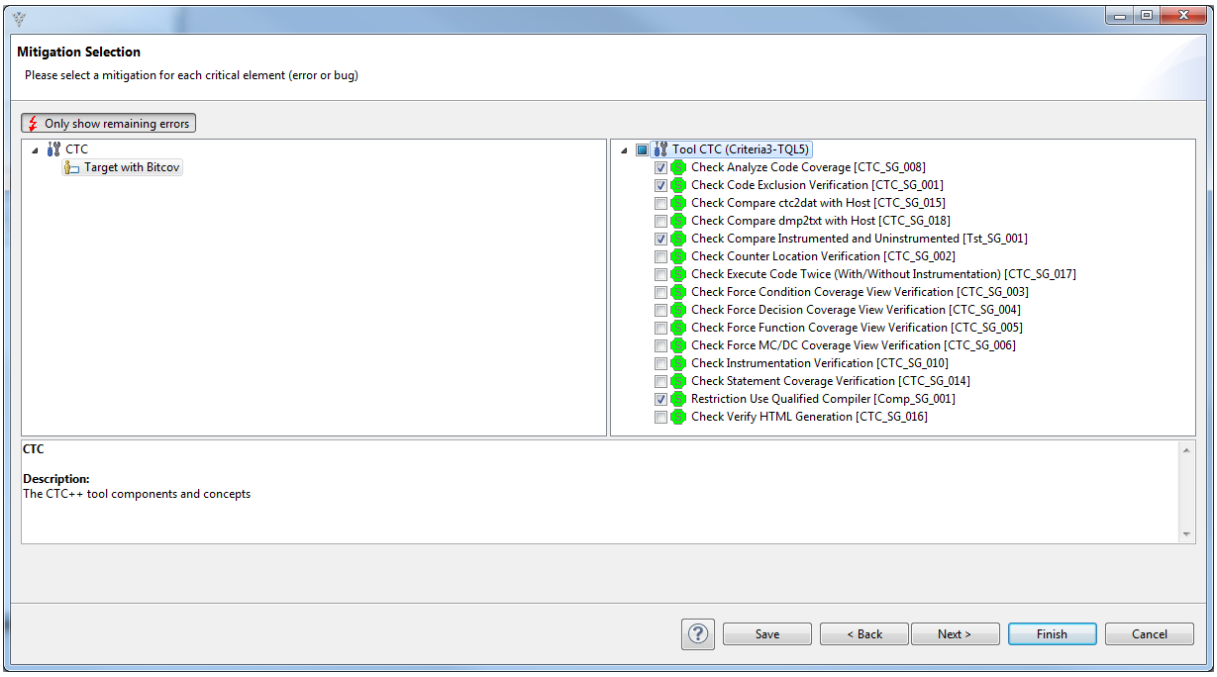


Figure 15: Mitigation Selection Page (remaining errors)

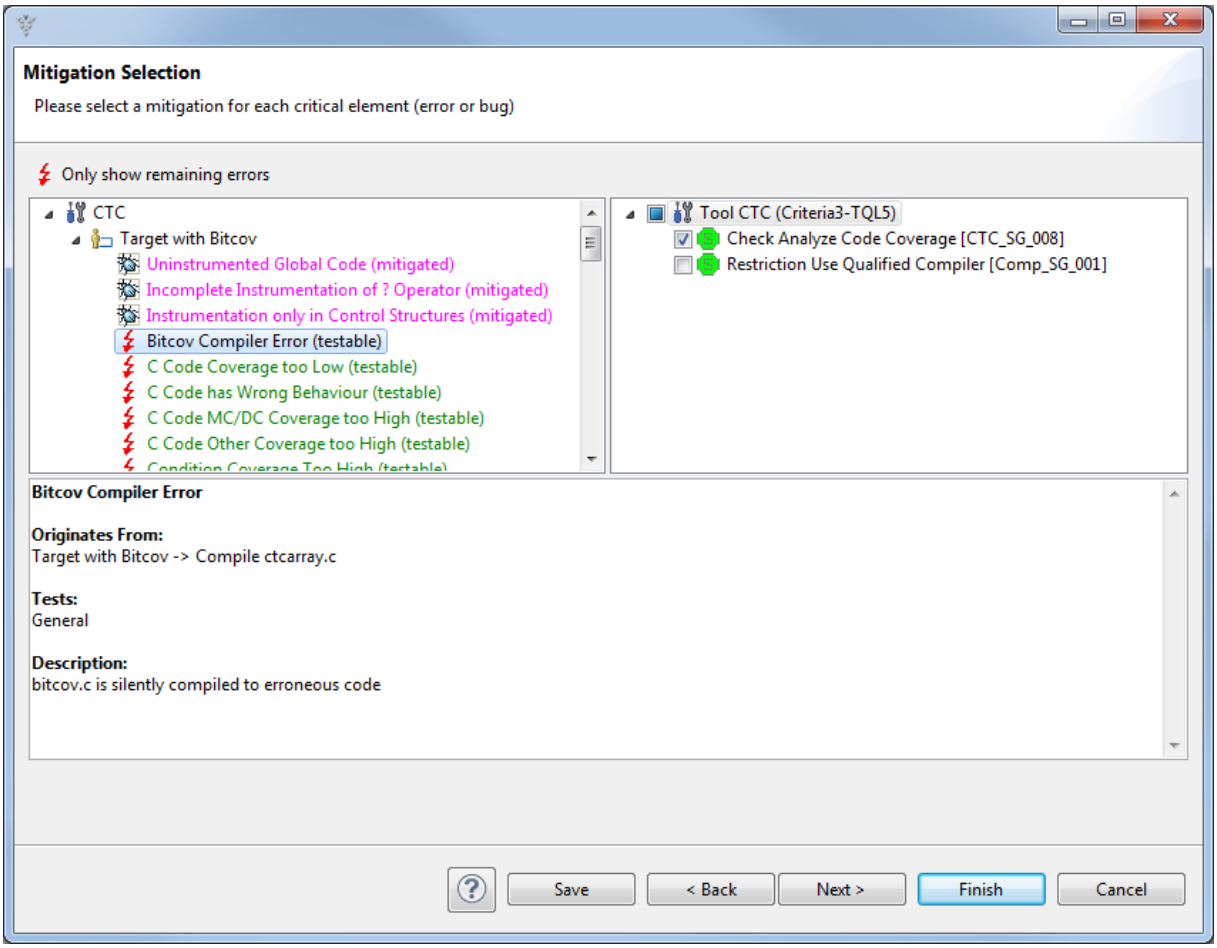


Figure 16: Mitigation Selection Page (all errors)

Navigation: At the bottom (below the information window) there is a navigation line, that allows to go to save the status ("Save") the next page ("Next >") or finish ("Finish") the qualification preparation. These buttons are only enabled if sufficient mitigations are selected for all potential errors that have no test cases.

Note that the qualification kit comes with a predefined use case that is already qualifyable. In Section 5.3.7 there are descriptions how this can be changed according to your selections to simplify the requalification with your selection.

5.3.5 Qualification Planning

The qualification mode contains a formalization of the qualification process the qualification can be planned using the QST. The plan will be generated into the Tool Qualification Plan as described in Section 5.3.6.
Tool qualification planning consists of three different elements that need to be specified:

1) Role assignment: There are qualification roles in the model, that need to be assigned to concrete persons that will be able to fulfill the role (see Figure 17). For that purpose they need to be selected in the tree and then the names can be edited.

2) Qualification Step Planning allows to select the start & end date for the qualification steps and to assign responsible roles to them, see Figure 18. Note that qualification steps can be hierarchic it is not necessary to plan each atomic step, but it is allowed to plan groups of steps. In this case the inherited information will be displayed.
3) Artifact Planning allows the user to select the path for the artifacts, see Figure 19.

Note that the planning shall be updated during qualification, for example to update the finished dates or the paths to the produced artifacts. By creating this information the qualification plan can be extended to a qualification report that documents all steps of the project.

All planned information will be contained in the tool qualification plan, that is extended to the tool qualification report as described in the plan.

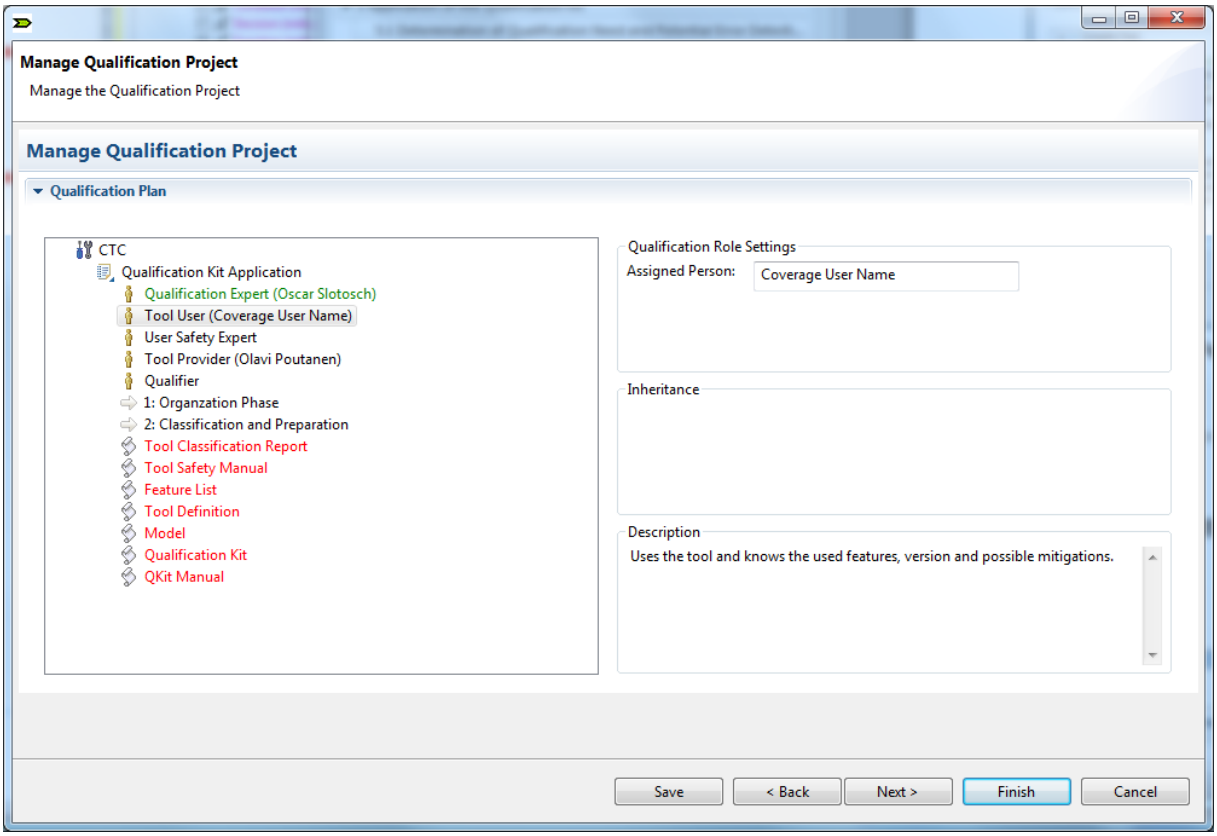


Figure 17: Qualification Planning - Role Assignment

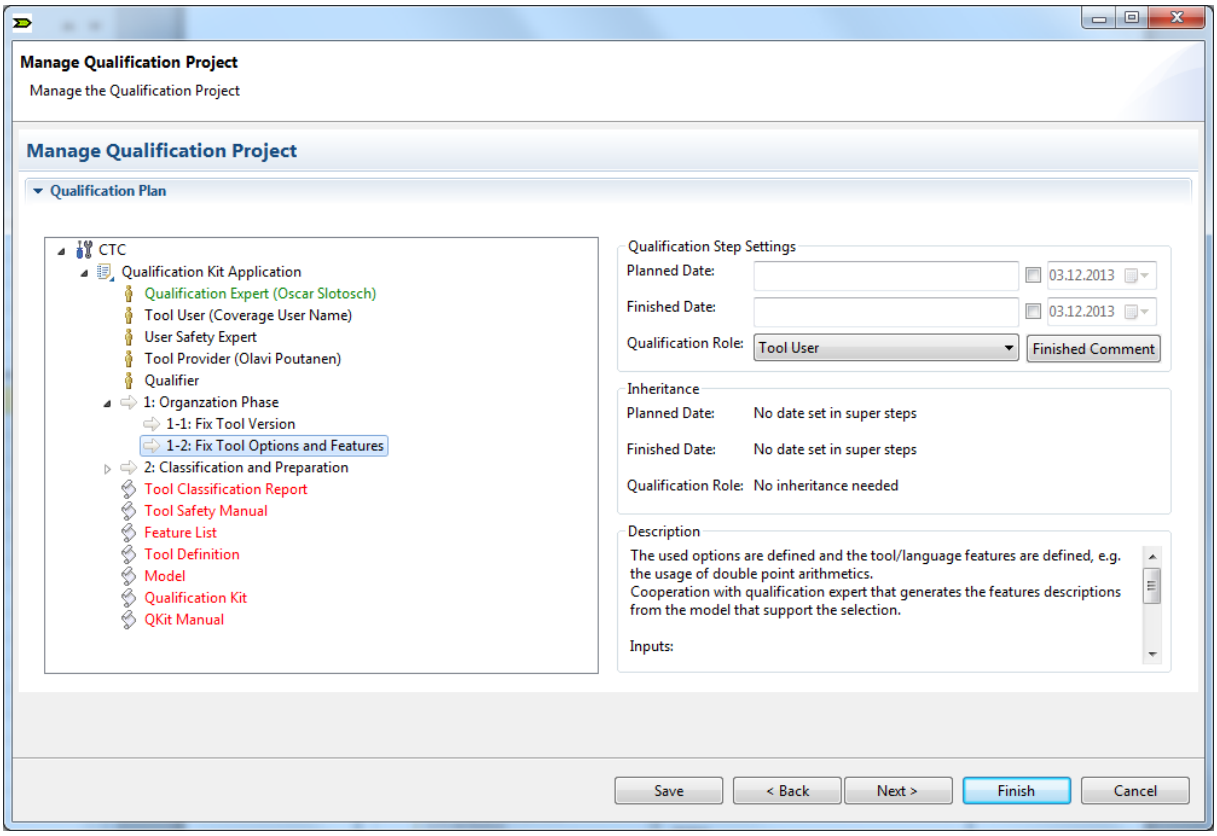


Figure 18: Qualification Planning - Step Planning

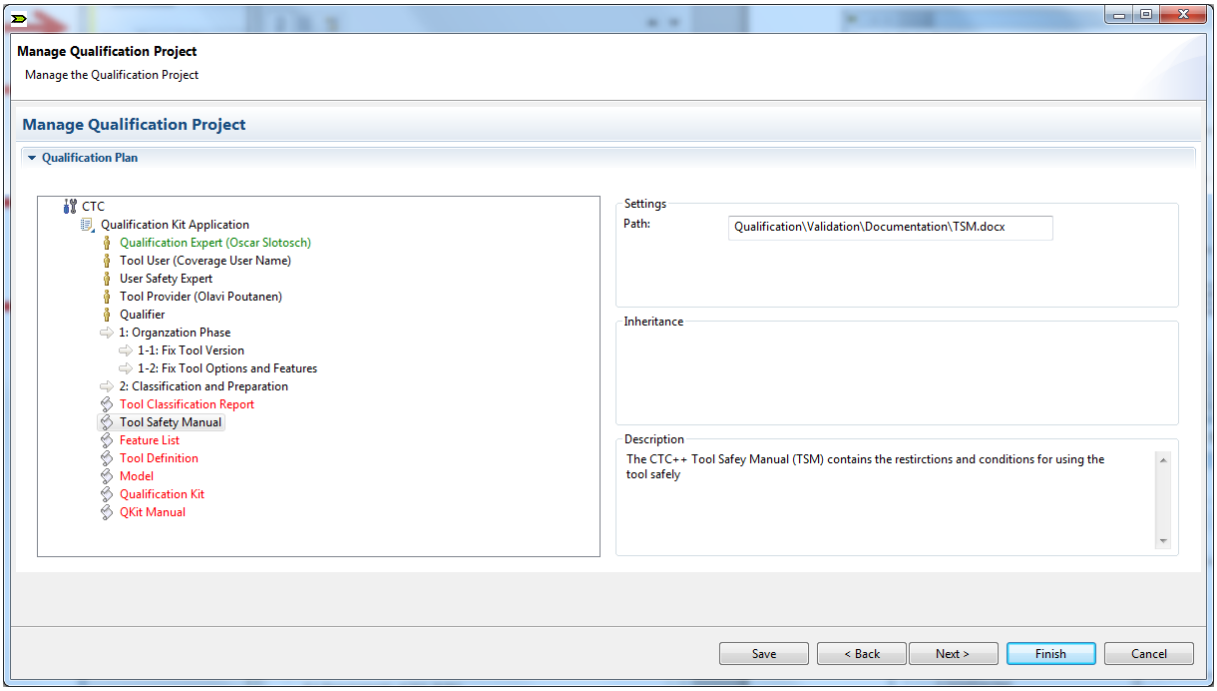


Figure 19: Qualification Planning - Artifact Planning

5.3.6 Generation of Qualification Documents

The qualification document generates the following documents depending on the selected model

- Tool Classification Report

- Tool Qualification Plan (if tests need to be executed)
- Tool Safety Manual
- Test Plan (if tests need to be executed)[3]

The other documents mentioned in the documentation structure (see section 4), do not depend on the method. The verification and verification report of the test cases is currently not generated for this qualification kit, since it is still in work. Therefore all available test cases shall be executed.

The qualification tool shows a summary page (after the configuration of the use case) with some statistics and the paths of the generated and copied documents. The paths are all subdirectories of the chosen qualification directory. Figure 20 shows an example:

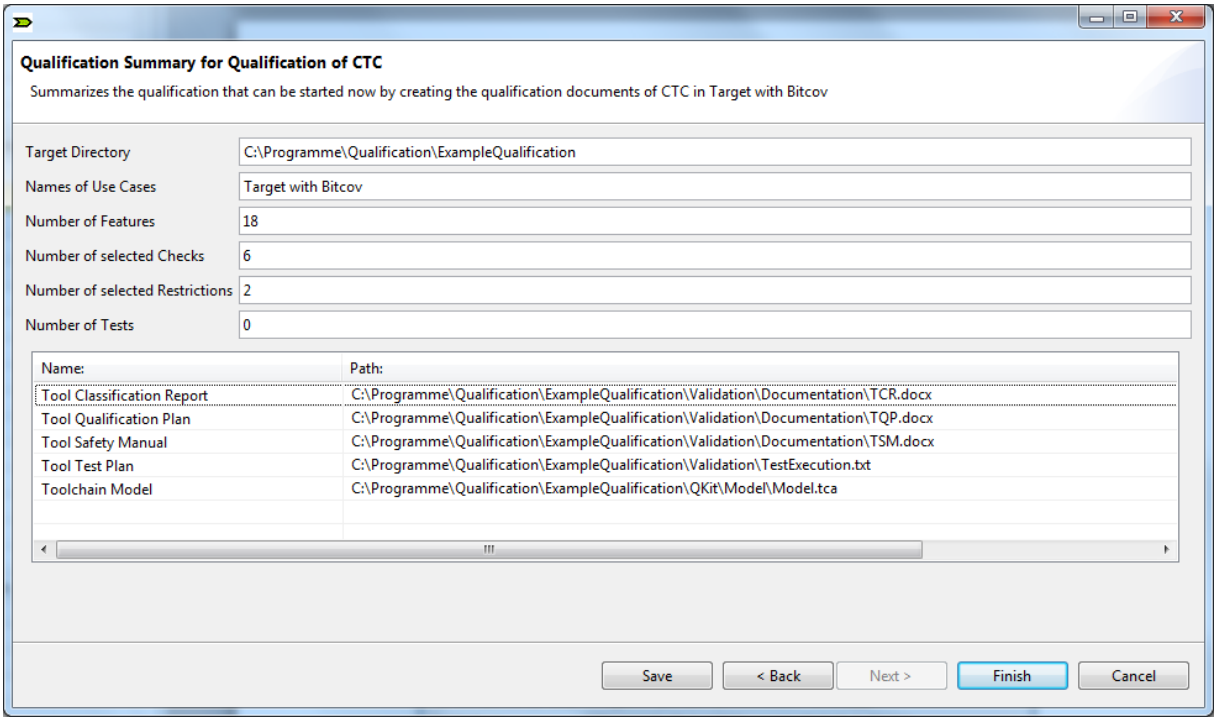


Figure 20: Qualification Summary with Paths to Generated Documents

If the "Finish" button is pressed the generation of the documents starts.

If the generation of the documents is finished the qualification support tool shows as depicted in Figure 21.

[3] Note currently there are no test cases in the Qualification Kit such that these documents do not need to be considered.

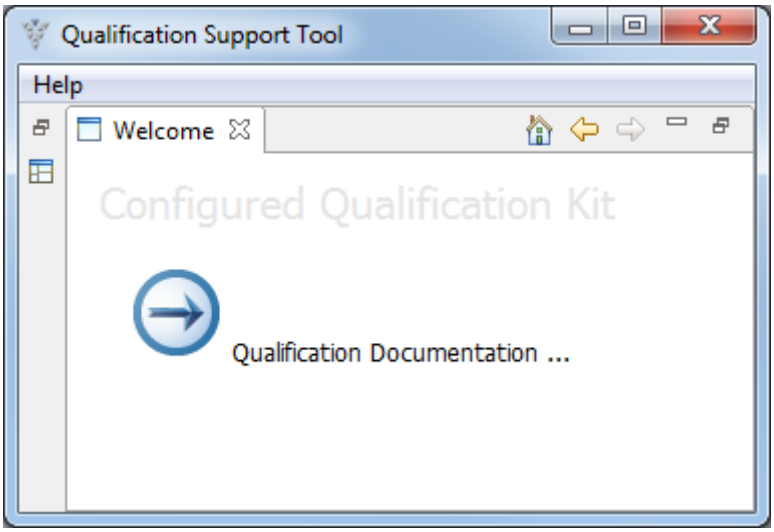


Figure 21: Finished Qualification Message

By clicking to the Qualification Documentation link the overview page of the material is shown (see Figure 22).

The documents can be opened by clicking on them (or via the file system).

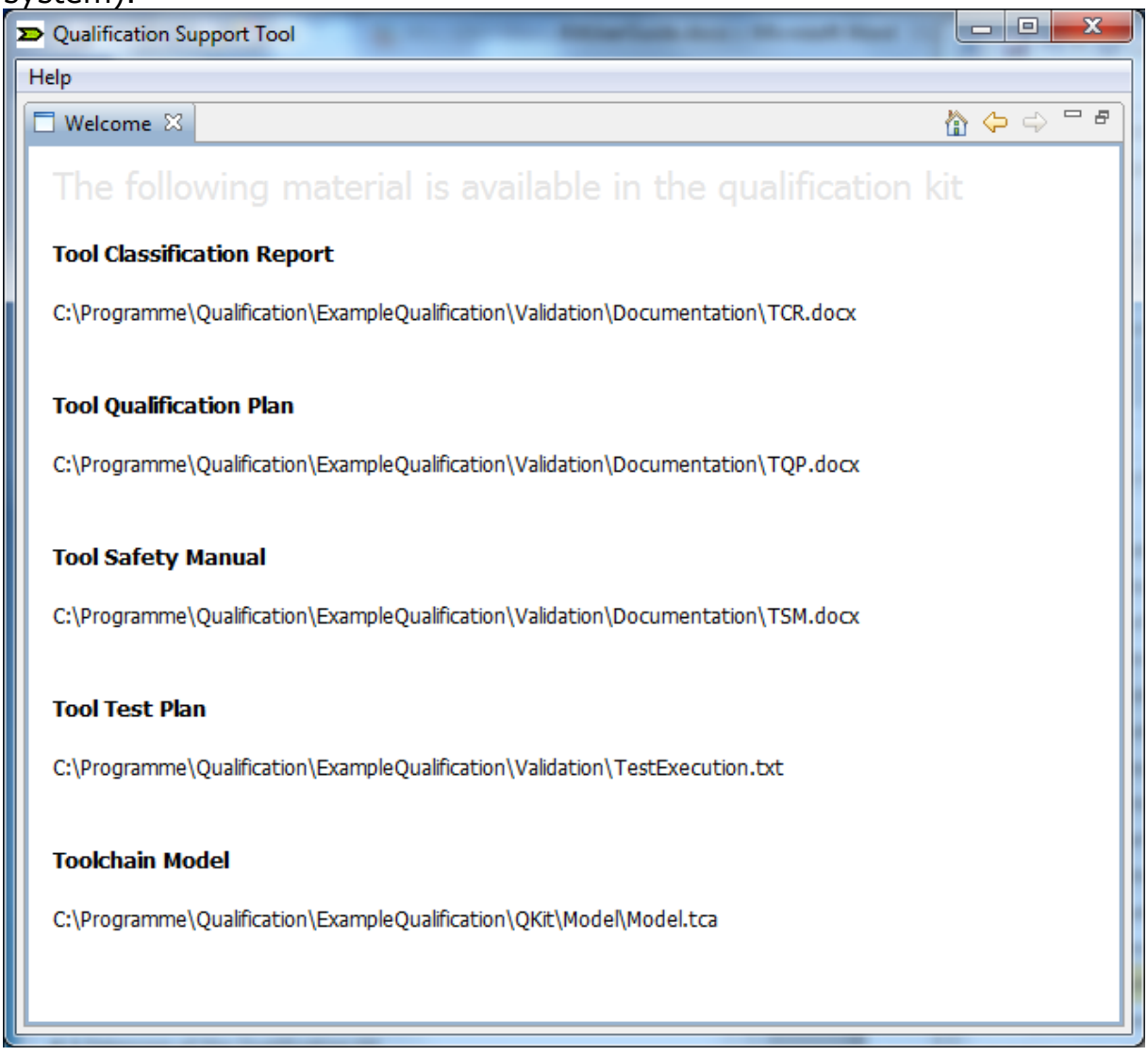


Figure 22: Generated Documents Overview

5.3.7 Customization

The Customization of the Qualification kit can be done in two places:

- The qualification kit: This is for all qualifications
- The Qualification directory: This is for the concrete qualification

The qualification support tool copies the qualification material from the qualification kit to the qualification directory and the uses it.

The qualification kit (QKit) is stored in the tool directory within a subdirectory called <plugins/MyModel> (see Figure 23). This contains a documentation directory with the documentation and the templates.

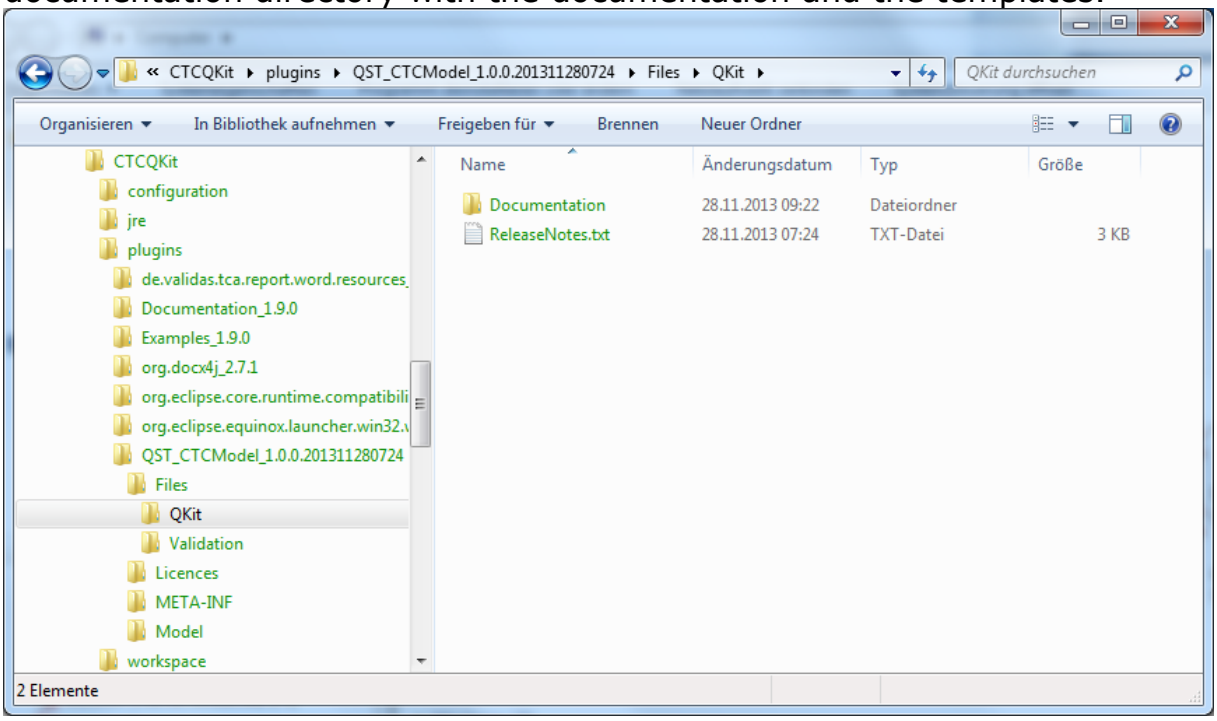


Figure 23: Qualification Kit in the Qualification Support Tool

After the qualification the documents are copied to the qualification target directory. This has a similar structure, especially it also contains the QKit (see Figure 24).

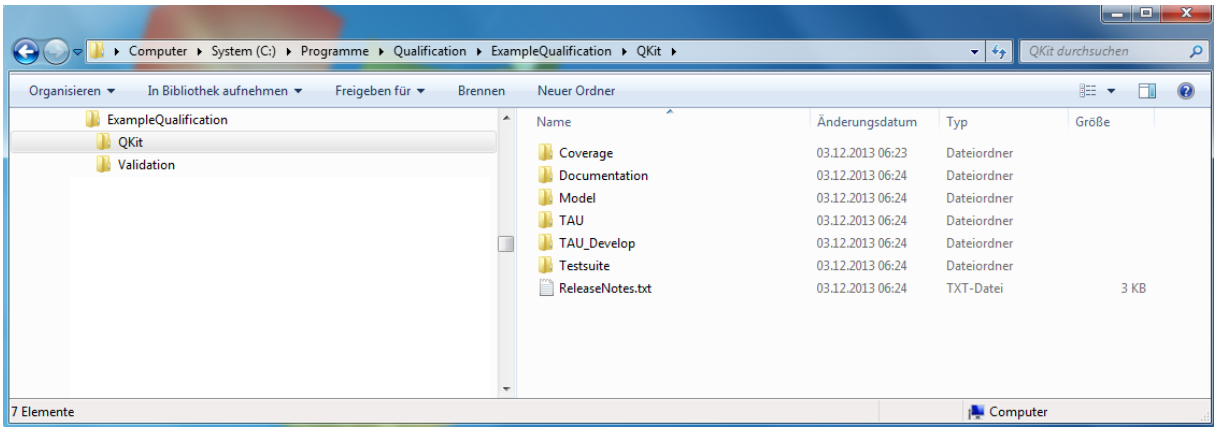


Figure 24: Qualification Kit in the Qualification Target Directory

Note that documents that already do exist in the Qualification target are not copied again. Especially if the qualification kit is changed, the qualification target has to be cleared in order not to work with the old copies of the templates.

The qualification support tool creates a model of the specified qualification configuration and stores it in the qualification target.
The stored model can be used to integrate the Testwell CTC++ into a development tool chain modeled with the TCA tool, or it can be used as a starting point for further qualifications.

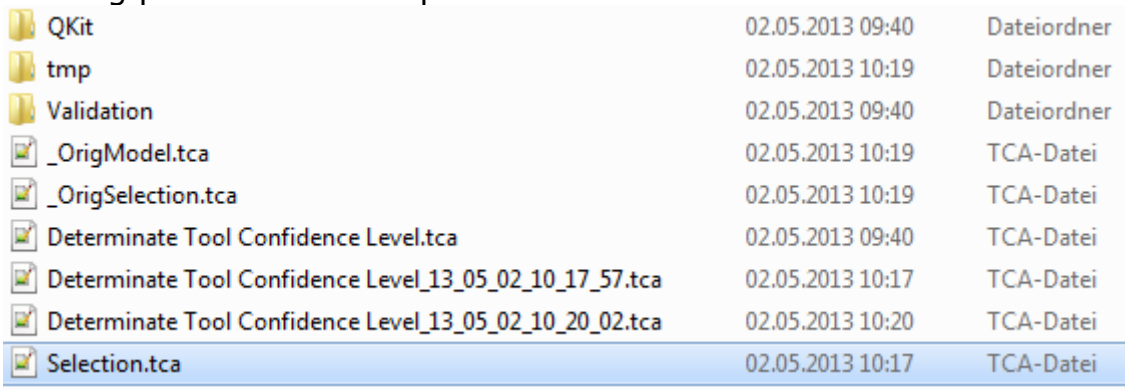


Figure 25: Qualification Configuration Examples

5.3.8 Licenses & Liability

The tool is a product of Validas AG. It has been developed using Eclipse and POI and docx4j. The licenses of these components are:

- Eclipse: EPL: http://www.eclipse.org/legal/epl-v10.html
- POI: Apache License Version 2.0: http://www.apache.org/licenses/ [4]
- docx4j: Apache License Version 2.0

They are distributed in the License directory of the tool. Validas has granted Verifysoft the right to copy and distribute the tool.

[4] Note the Apache Licence 2.0 is distributed with this product in the jar file of the plugin de.validas.excelinterface, which is found in the plugins directory of the TCA.

Validas AG does not take any guarantee for the functionality of the tool. Since all produced documents have to be reviewed this tool has no confidence need and can be used unqualified.

VALIDAS AG AND ITS AFFILIATES MAKE NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. VALIDAS AG AND ITS AFFILIATES SHALL NOT BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATED TO THIS PUBLICATION OR THE INFORMATION CONTAINED IN IT, EVEN IF VALIDAS AG AND ITS AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

6 Extension of the Qualification Kit

The tool qualification kit for the Testwell CTC++ can be extended from the user with the following elements

- New *Feature* elements for new functions that are either new or currently not considered in the qualification kit
- New safety guidelines
- New *Test* elements for new test cases

Note that the consideration of a new use case, based on the existing qualification kit is not considered as extension and is described in Section 5. Every extension of the kit consists of maximal four steps that need to be executed and are described in the following sections:

1) Extending only the Model, see Section 6.1,
2) Validating the Model, see Section 6.2,
3) Extending the Test Suite, see Section 6.3 and
4) Validating the Test Suite, see Section 6.4.

Note that it is possible to extend the qualification kit only by extending the model, e.g. by adding a new safety guideline or new features. This requires a validation of the model, but not necessarily the creation of new test cases. If new test cases are added they also need a validation and usually also a representation in the model. The structure of the model is explained in [TCA_UM].

6.1 Extending only the Model

The model can be extended by new functions and new safety guidelines. These additions need to be validated as described in Section 6.2.

6.1.1 Adding new Features

To add a new function a *Feature* element has to be created. This is done in several steps that are described in the following subsections

1) Structure Creation, see Section 6.1.1.1,
2) Structure Characterization, see Section 6.1.1.2,
3) Error Modeling, see Section 6.1.1.3 and
4) Mitigation Modeling, see Section 6.1.1.4.

Section 6.2.1 shows how the new *Feature* elements shall be validated.

6.1.1.1 Structure Creation

Creating the structure for new *Feature* elements can be either in the *Tool* element or in another *Feature* element. After inserting the element the properties need to be set for the new element:

- *Name*: Should be unique in the tool chain due to the images
- *Description* / *Long Description*

If the *Feature* element is added to another *Feature* it automatically inherits the potential *Error* elements of this *Feature*. Furthermore the input outputs are inherited.
Next step is to model the artifacts of *Feature* elements. This can either be the inherited ones, or to assign new artifacts. The artifacts can be assigned to the list of *Inputs, Outputs* and *Input Outputs*. The assigned artifacts can be either from the existing artifacts of the Testwell CTC++ model, or new *Artifact* elements can be created in the *Tool Chain* element. This finalizes the structure model.

6.1.1.2 Structure Characterization

The analysis model requires characterizing the new feature by assigning *Tool Attribute* elements to it. In the case that new *Artifact* elements have been created in the *Tool Chain* they need also a characterization using *Artifact Attributes*.

6.1.1.3 Error Modeling

Starting with the *Expand Attribute* action the TCA creates *Derived Error* elements in the *Feature* element that have to be subsumed by concrete *Error* elements that have to be created as new elements in the *Feature*. Subsuming the errors is done by adding the derived errors to the list *Subsumes*. If *Derived Error* elements from *Artifact Attribute* elements are subsumed the *Artifacts* have to be added to the *Artifacts* list of the *Error* element. After subsuming the errors they can be removed. This is automatically done from the *Expand Attribute* action or can be done manually.

6.1.1.4 Mitigation Modeling

For every new potential *Error* element there should be one mitigation element in a safety guideline assigned, such that the user can mitigate the error by applying the safety guideline. If no appropriate guideline is available new guidelines can be created as described in Section 6.1.2. If there is no possibility at all to mitigate the error, than there should be assigned some test cases as described in Section 6.3.

6.1.2 Adding new Safety Guidelines

The creation of safety guidelines is done in the following steps:

1) Creation of a Feature element in the *Feature Safety Guidelines* with
- *Name*: starting *with* the Prefix *SG_*
- *Description*: containing a short description
- *Virtual*: value set to *true*

2) Creation of a *Check* / *Restriction* element in the new safety guideline *Feature* with
- *Name*: characterizing the mitigation
- *Description* / *Long Description*

- *Detection / Avoidance Probability* selected
- *Comment* explaining the probability if it is *HIGH*

3) Assignment of the *Check* / *Restriction* elements to the existing *Error* elements of Testwell CTC++.

Section 6.2.2 shows how the new safety guideline shall be validated.

6.2 Validating the Model

The model can be extended by new features and new safety guidelines. As described in Section 6.1. This section describes how to validate the extensions.

Validation is done in two steps:

1) TCA: The TCA has a validation function that detects some syntactical inconsistencies in the model. All found validation errors shall be analyzed, and if possible removed to reduce the overhead to remove them after they have found during review of the generated documents.
2) Manual review: The steps for manually reviewing the created model depend on the created elements and are described in the following sub-sections. The manual reviews can be done based on Excel tables for the different views that are exported from the TCA for that purpose.

6.2.1 Validating new Features

As described in Section 6.1.1 the creation of new *Feature* elements is done in the following steps:

1) Structure Creation, see Section 6.1.1.1,
2) Structure Characterization, see Section 6.1.1.2,
3) Error Modeling, see Section 6.1.1.3 and
4) Mitigation Modeling, see Section 6.1.1.4.

They are all validated from the user in the following checks:

1) Review of TA-Matrix, see Section 6.2.1.1 validates the Structure Creation,
2) Review of Attributes Assignment, see Section 6.2.1.2 validates the Structure Characterization, and
3) Review of Error-Assignments, see Section 6.2.1.3 validates the Error Modeling and the Mitigation Modeling.

6.2.1.1 Review of TA-Matrix

The Tool-Artifact Matrix (TA-Matrix) describes how the tools are connected. The TA-Matrix can be exported from the TCA by selecting the *Tool Chain* element and starting the action *Export -> Excel Tool-Artifact Matrix*.

An example for a TA-Matrix can be found in Figure 26. Note that if there are some changes detected during the review, they can be imported into the TCA tool using the action *Import-> Excel Tool-Artifact Matrix* of the *Tool Chain* element.

	Artifact Display Output	Artifact Excel File	Artifact Model	Artifact Overall project plan	Artifact Review Protocol	Artifact Safety Manual	Artifact Safety plan	Artifact Tool Evaluation Report	Artifact User Input	Artifact Word Document
ISO 26262 Reviews					w					
ISO 26262 Reviews / UseCase Confirmation Review Of TCLs			r			r		r		
Process Checker										
Process Checker / UseCase Validate Process				r,w			r,w			
Tool Tool Chain Analyzer										
Tool Tool Chain Analyzer / Feature Compute Tool Confidence Level	w	w	r,w						r	w
Tool Tool Chain Analyzer / Feature Cost Model	w	r,w	r,w						r	
Tool Tool Chain Analyzer / Feature EMF	w		r,w						r	
Tool Tool Chain Analyzer / Feature Excel Interface		r,w	r,w						r	
Tool Tool Chain Analyzer / Feature Generate Word (docx)			r						r	w
Tool Tool Chain Analyzer / Feature Model Validation	w		r						r	
Tool Tool Chain Analyzer / Feature Safety Guidelines										
Tool Tool Chain Analyzer / Feature Safety Guidelines , SG_Avoid Feature										
Tool Tool Chain Analyzer / Feature Safety Guidelines , SG_Use Review Checklist										
Tool Tool Chain Analyzer / Feature Xml Interface			r,w						r	
Tool Tool Chain Analyzer / UseCase Cost Calculation										
Tool Tool Chain Analyzer / UseCase Create Model				r			r			
Tool Tool Chain Analyzer / UseCase Determinate Tool Confidence Level				r		w	r	w		
Tool Tool Chain Analyzer / UseCase Generate Tool Classification Report				r		r,w		w		
Tool Tool Chain Analyzer / UseCase Review Model				r	w	r,w	r			

Figure 26: Tool-Artifact Matrix of the TCA

6.2.1.2 Review of Attributes Assignment

The attribute assignment describes how the *Attribute* elements of the general error model are assigned to the model elements to characterize them and to derive errors. *Artifact Attribute* elements can be assigned to *Artifact* elements and *Tool Attribute* elements can be assigned to *Tool* and *Feature* elements. The attribute assignment can be exported from the TCA by selecting the *Tool Chain* element and starting the action *Export -> Excel Tool Attribute*.

An example for an attribute assignment can be found in Figure 27. Note that if there are some changes detected during the review, they can be imported into the TCA tool using the action *Import-> Excel Tool Attribute* of the *Tool Chain* element.

Attribute -> Tool / Feature / Use Case / Artifact	ArtifactAttribute Data_File	ArtifactAttribute Data_File_Syntax	ArtifactAttribute Data_File_Syntax_Model	ArtifactAttribute Data_File_Table	ArtifactAttribute Data_File_Text	ArtifactAttribute Data_Interaction	ToolAttribute Fcn_Algorithm	ToolAttribute Fcn_Algorithm_DeEncode	ToolAttribute Fcn_Behaviour	ToolAttribute Fcn_Behaviour_Calculator	ToolAttribute Fcn_Behaviour_Transformation	ToolAttribute Fcn_Resource_CPU	ToolAttribute Fcn_Specification	ToolAttribute Fcn_Variants	ToolAttribute Fcn_Variants_Options
Artifact Display Output					X	X									
Artifact Excel File	X	X		X											
Artifact Model	X		X												
Artifact Overall project plan	X				X										
Artifact Review Protocol	X	X			X										
Artifact Safety Manual	X	X			X										
Artifact Safety plan	X	X													
Artifact Tool Evaluation Report	X	X			X										
Artifact User Input						X									
Artifact Word Document	X	X			X										
Tool Tool Chain Analyzer / Feature Compute Tool Confidence Level							X			X			X	X	
Tool Tool Chain Analyzer / Feature Cost Model															
Tool Tool Chain Analyzer / Feature EMF							X		X		X			X	
Tool Tool Chain Analyzer / Feature Excel Interface								X	X		X				X
Tool Tool Chain Analyzer / Feature Generate Word (docx)											X	X	X	X	
Tool Tool Chain Analyzer / Feature Model Validation							X		X				X		

Figure 27: Example of Attribute Assignment of the TCA

6.2.1.3 Review of Error-Assignments

The assignment of *Check* and *Restriction* elements to Error elements is reviewed in the most important review of the model. It contains a list of all *Error*, *Check* and *Restriction* elements with the following information:

- *Tool*: the tool in which the element occurs,
- *Use Case*: the tool in which the element occurs,
- *Type*: The type of the element (*Error*/*Check*/*Restriction*),
- *Assumption*: flag indicating whether the element is implemented
- *Element*: the name
- *Inherited from*: the name of the features where the element is inherited from
- *Derived from*: the name of the general error model element where the element is derived from
- *Subsumes*
- *Probability*
- *Description*
- *Linked to*: relation between Check/Restriction to *Error* elements

The following information shall be filled out from the reviewer:

- Reviewed by
- Is probability OK/NOK?: This includes reviewing the assignment of *Error* elements to
- For *Checks/Restrictions*: Is currently performed true/false?
- Comments: further errors/checks restrictions, etc.?

This main review can be exported from the TCA by selecting the *Tool Chain* element and starting the action *Export -> Excel Review*. Figure 28 shows an example of an exported review form for the error assignments.

Tool	Usecase	Type	Assumption	Element	Inherited from	Derived from	Subsumes	Probability	Description	Linked to	Reviewed by	Is probability OK/NOK?	For Checks/Restrictions: Is currently performed true/false?	Comments: further errors/checks/restrictions, etc.?
Process Checker	Validate Process	Restriction	false	Consistent Process				HIGH	This ensures that the process	Error:Tool Tool Chain Analyzer / Feature				
ISO 26262 Reviews	SG_Confirmation Revi	Check	true	Detect Wrong TCL				HIGH	An error in the TCL computatic	Error:Tool Tool Chain Analyzer / Feature (				
Tool Chain Analyzer	Cost Calculation	Error	false	Wrong Cost Compute	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	LOW	The costs for the tools are con	Restriction:Tool Tool Chain Analyzer / Fe				
Tool Chain Analyzer	Cost Calculation	Error	false	Any EMF Error	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	LOW	Any error that can occur in EM					
Tool Chain Analyzer	Cost Calculation	Error	false	Wrong Export	Tool Tool Chain Analy		Fcn_Algorithm_DeEncode.D	HIGH	The excel file does not contain	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Cost Calculation	Error	false	Wrong Import	Tool Tool Chain Analy		Data_File_Syntax_Model.En	HIGH	The model is created wrongly.	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Review Model	Error	false	Model Not Adequate			Data_Interaction.No Interacti	HIGH	An important issue as not bee	Check:Tool Tool Chain Analyzer / Feature				
Tool Chain Analyzer	Review Model	Error	false	Any EMF Error	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	LOW	Any error that can occur in EM					
Tool Chain Analyzer	Review Model	Error	false	Wrong Error Reported	Tool Tool Chain Analy			LOW	The model validation might rep	Restriction:Tool Tool Chain Analyzer / Fe				
Tool Chain Analyzer	Review Model	Error	false	Process Inconsistenth	Tool Tool Chain Analy		Data_File.Defect File,Data_F	HIGH	The process might be inkonsis	Restriction:Process Checker / UseCase '				
Tool Chain Analyzer	Review Model	Error	false	Wrong Export	Tool Tool Chain Analy		Fcn_Algorithm_DeEncode.D	HIGH	The excel file does not contain	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Review Model	Error	false	Wrong Import	Tool Tool Chain Analy		Data_File_Syntax_Model.En	HIGH	The model is created wrongly.	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Review Model	Check	false	Review Checklist	Tool Tool Chain Analy			HIGH	The model review can be perfo	Error:Tool Tool Chain Analyzer / UseCase				
Tool Chain Analyzer	Generate Tool Classifi	Error	false	Document Generated	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	HIGH	Document does not fit to the n	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Generate Tool Classifi	Error	false	Any EMF Error	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	LOW	Any error that can occur in EM					
Tool Chain Analyzer	Generate Tool Classifi	Error	false	Wrong TCL Computed	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	HIGH	The TCL is computed wronly, (	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Determinate Tool Cont	Error	false	TCL Wrongly Shown			Data_Interaction.No Interacti	HIGH	TCL is computed correctly but	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Determinate Tool Cont	Error	false	TCL Wrongly Written			Data_File_Syntax_Model.En	HIGH	TCL is computed or written wr	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Determinate Tool Cont	Error	false	Wrong TCL Computed	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	HIGH	The TCL is computed wronly, (	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Determinate Tool Cont	Error	false	Any EMF Error	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	LOW	Any error that can occur in EM					
Tool Chain Analyzer	Create Model	Error	false	Any EMF Error	Tool Tool Chain Analy		Data_File_Text.Defect Text,[	LOW	Any error that can occur in EM					
Tool Chain Analyzer	Create Model	Error	false	Wrong XML Export	Tool Tool Chain Analy			HIGH	The xml file does not contain t	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Create Model	Error	false	Wrong XML Import	Tool Tool Chain Analy		Data_File.Defect File,Data_F	HIGH	The model is created wrongly.	Check:ISO 26262 Reviews / UseCase SC				
Tool Chain Analyzer	Create Model	Error	false	Wrong Error Reported	Tool Tool Chain Analy			LOW	The model validation might rep	Restriction:Tool Tool Chain Analyzer / Fe				
Tool Chain Analyzer	Create Model	Error	false	Process Inconsistenth	Tool Tool Chain Analy		Data_File.Defect File,Data_F	HIGH	The process might be inkonsis	Restriction:Process Checker / UseCase '				

Figure 28: Example Review Form for the TCA

6.2.2 Validating new Safety Guidelines

The validation of safety guidelines is done with the same review as the creation of new features, see Section 6.2.1.3.

6.3 Extending the Test Suite

The test suite can be extended top-down and bottom-up way. In the top-down approach the missing tests are identified in the model and requirements for the test are implemented. In the bottom-up approach new test cases are added and classified according to the model. The following sub-sections describe the two approaches.

6.3.1 Top-Down Approach

The top-down approach for the creation of test cases starts from the model that has some unmitigated errors (for example due to a de-selection of a safety guide). The task for extending the test suite is then to provide tests and explanations to demonstrate that these unmitigated errors cannot occur in the tool.

The TCA supports the top-down approach by the function to create the missing tests. To start this function a *Feature*, *Use Case* or a *Tool* element has to be selected in the browser and the action "*Generate Missing Test*" creates a test model structure that conforms to the test models in [TCA_UM] and describes the requirements to the tests (see Figure 29)

including a long description that explains the error by listing all subsumed errors of the general error model (see Figure 30).

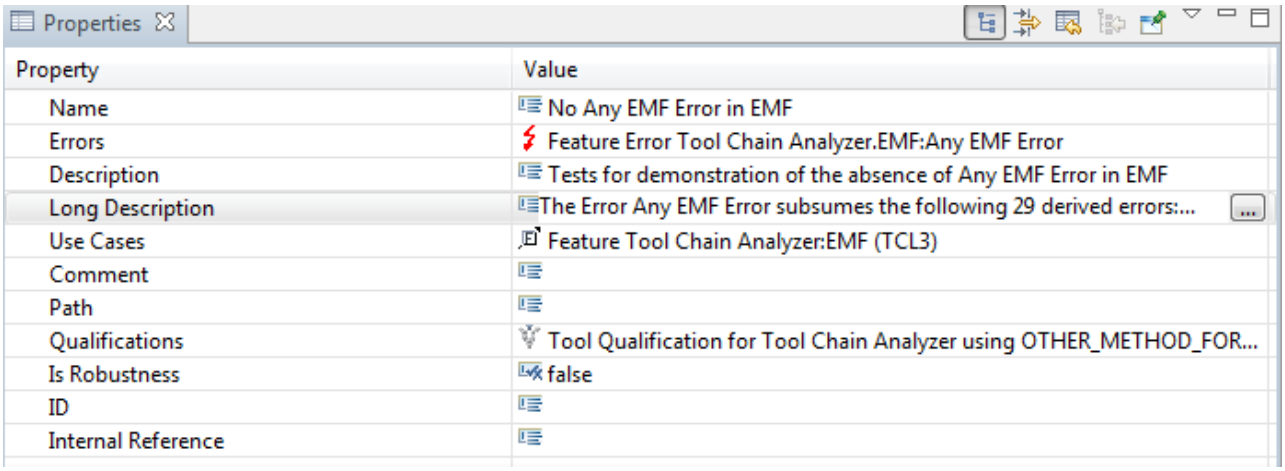


Figure 29: Properties of a Generated Missing Test

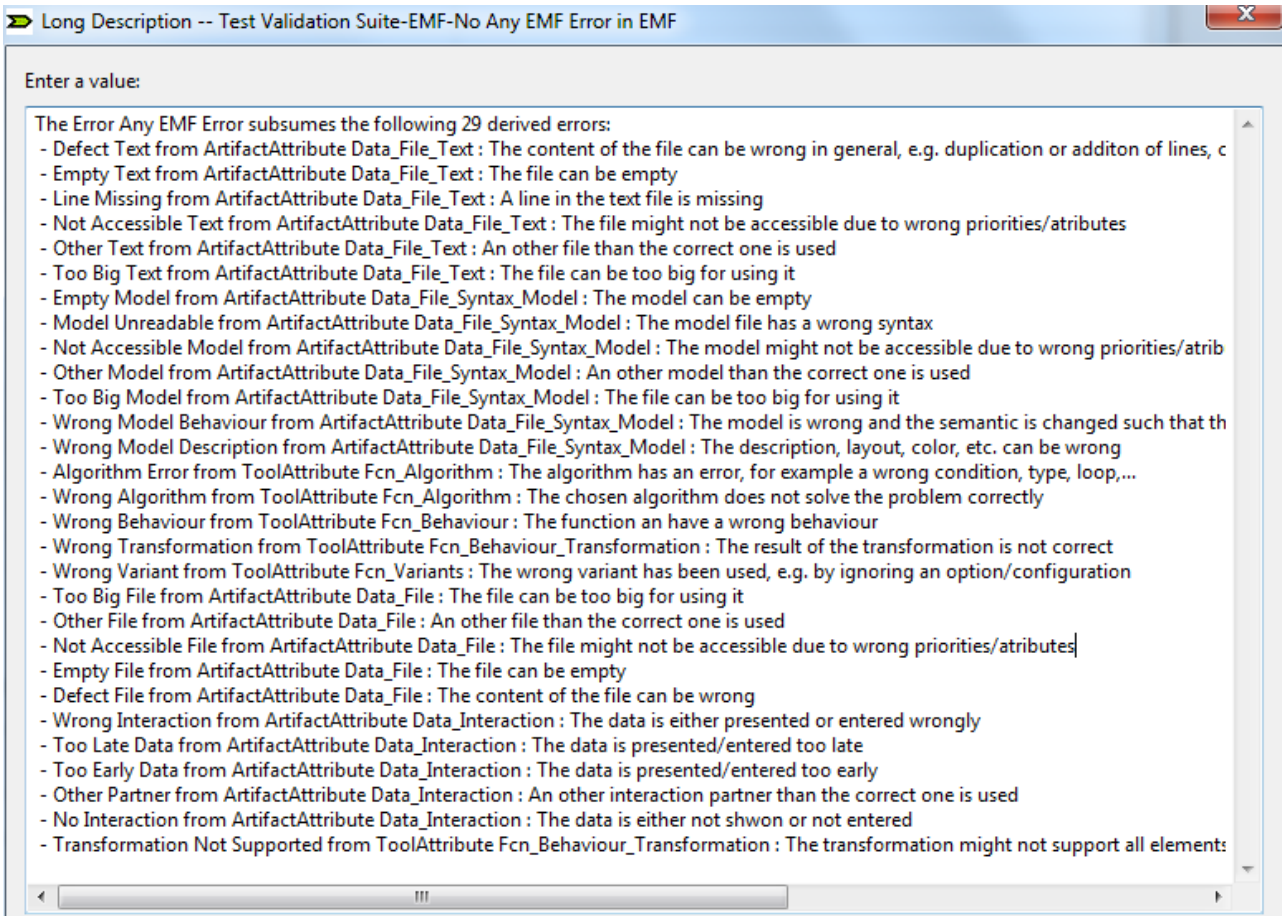


Figure 30: Long Description of a Generated Missing Test

Note that the generated missing tests are only requirements and have no corresponding implementation. Hence the *Path* property in these tests is not set to a value (see Figure 29). For the implementation of tests a directory structure can be generated with the TCA, such that the implementer of the test cases requires only to put the test cases into the generated directories and to provide an explanation how the absence of

the errors is shown by the test. This can be done using the action *Create File Structure* that can be triggered for every *Test* element. This action also creates the values for the *Path* properties into the model.

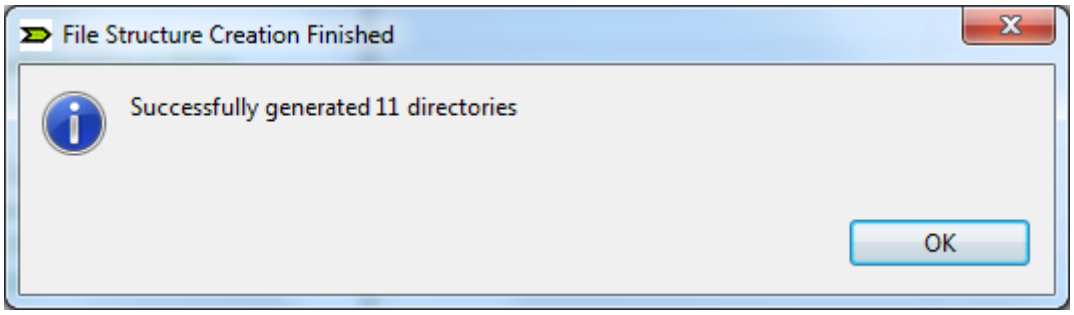


Figure 31: Message from the Generation of File Structure

Of course the generated directories are empty (but the model successfully validates and conforms to all requirements). Therefore the validation of the test suite is a very important step which is described in Section 6.4.

6.3.2 Bottom-Up Approach

The bottom-up approach for the extension of the test suite is to add further test cases to the model. This can be done by importing the test case structure (i.e. the directory structure) into the model and manually linking it to the Feature and Error elements they belong.

The import of the test structures can be done using the *Import Test Structure* action for a selected *Tool* element in the TCA. The result is a message as depicted in Figure 32.

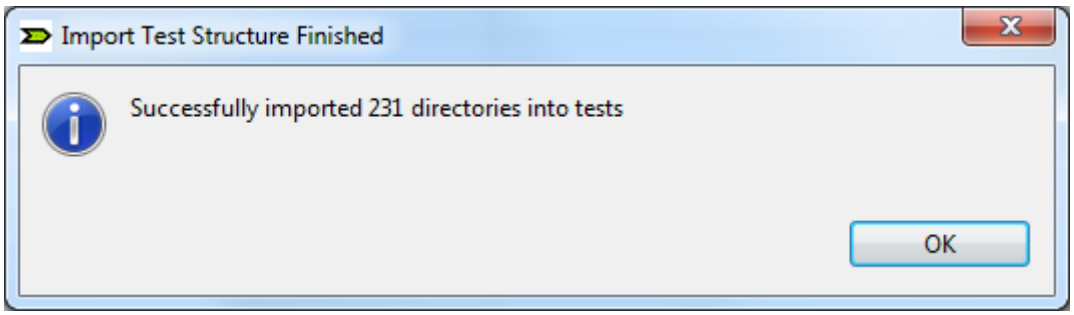


Figure 32: Importing Test Structures into the Model

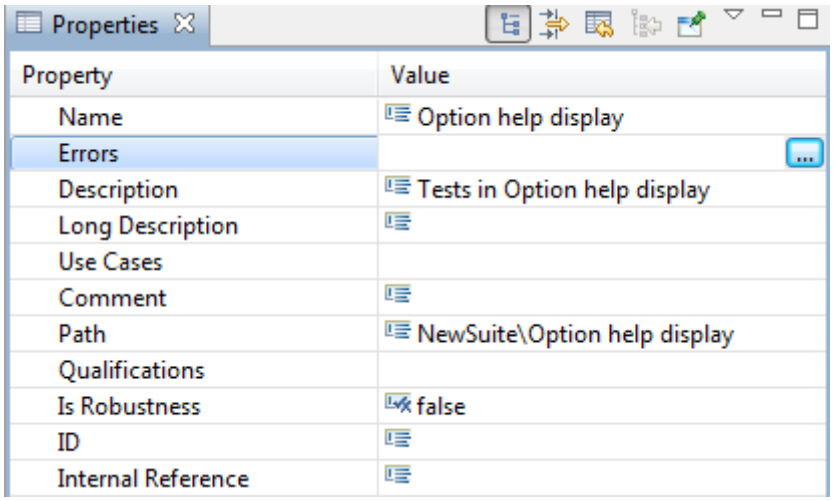


Figure 33: Properties of Imported Test Elements

The imported *Test* elements have set the following properties (see Figure 33): *Name*, *Description* and *Path*. The following steps need to be done:

1) Assign the *Test* element to the *Error* elements that can be detected with the test (using the *Errors* list),
2) Assign the *Test* element to the *Feature* elements that are tested with the test (using the *Use Cases* list) and
3) Describe an argumentation why the imported tests suffice to show the absence of the assigned errors (using *Long Description*).

6.4 Validating the Test Suite

The validation of the test suite shall ensure that the implemented test cases satisfy their specifications in the model, which is to show the absence of the errors in the use cases of the tool. For that purpose a checklist is generated that can be used as a basis for the validation. The action *Export -> Excel Test Review* is used for that if a *Tool* element is selected. A message shows the number of exported review items (see Figure 34).

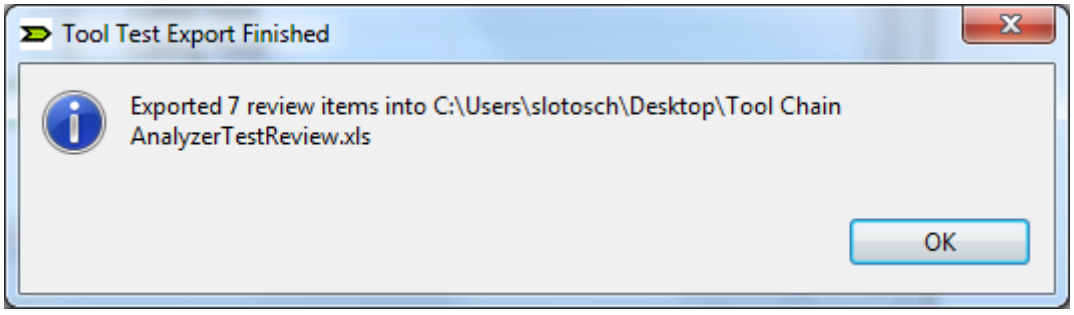
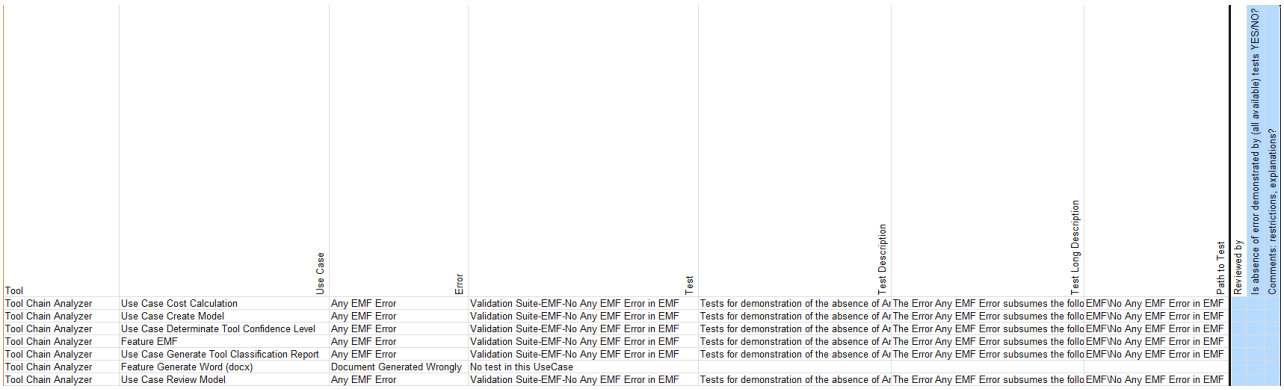


Figure 34: Export Message of Test Review

The reviews are structures according to the use cases to which they apply, such that some tests are listed several times. This is the case if the *Feature* elements are used several times and the *Feature* error can occur in several use cases. The effectiveness of the test has to be ensured for all considered use cases.



Tool	Use Case	Error	Test	Test Description	Test Long Description	Path to Test	Reviewed by	Is absence of error demonstrated by (all available) tests YES/NO?	Comments: restrictions, explanations?
Tool Chain Analyzer	Use Case Cost Calculation	Any EMF Error	Validation Suite-EMF-No Any EMF Error in EMF	Tests for demonstration of the absence of Ar	The Error Any EMF Error subsumes the follo	EMF\No Any EMF Error in EMF			
Tool Chain Analyzer	Use Case Create Model	Any EMF Error	Validation Suite-EMF-No Any EMF Error in EMF	Tests for demonstration of the absence of Ar	The Error Any EMF Error subsumes the follo	EMF\No Any EMF Error in EMF			
Tool Chain Analyzer	Use Case Determinate Tool Confidence Level	Any EMF Error	Validation Suite-EMF-No Any EMF Error in EMF	Tests for demonstration of the absence of Ar	The Error Any EMF Error subsumes the follo	EMF\No Any EMF Error in EMF			
Tool Chain Analyzer	Feature EMF	Any EMF Error	Validation Suite-EMF-No Any EMF Error in EMF	Tests for demonstration of the absence of Ar	The Error Any EMF Error subsumes the follo	EMF\No Any EMF Error in EMF			
Tool Chain Analyzer	Use Case Generate Tool Classification Report	Any EMF Error	Validation Suite-EMF-No Any EMF Error in EMF	Tests for demonstration of the absence of Ar	The Error Any EMF Error subsumes the follo	EMF\No Any EMF Error in EMF			
Tool Chain Analyzer	Feature Generate Word (docx)	Document Generated Wrongly	No test in this UseCase						
Tool Chain Analyzer	Use Case Review Model	Any EMF Error	Validation Suite-EMF-No Any EMF Error in EMF	Tests for demonstration of the absence of Ar	The Error Any EMF Error subsumes the follo	EMF\No Any EMF Error in EMF			

Figure 35: Example Test Review for the TCA Test Elements

The test review is the only review of the test implementation. It contains a list of all *Test* elements assigned to Use Case elements with following information:

- *Tool*
- *Use Case*,
- *Error*,
- *Test*,
- *Test Description*
- *Test Long Description*
- *Path to Test*

The following information shall be filled out from the reviewer:

- Reviewed by
- Is absence of error demonstrated by (all available) tests YES/NO?
- Comments: restrictions, explanations?

7 References

[DO330] RTCA. DO-330: Software Tool Qualification Considerations 1st Edition 2011-12-13.

[EN50128]: BS EN 50128:2011, Railway applications — Communication, signalling and processing systems — Software for railway control and protection systems, BSI Standards Publication

[IEC61508]International Electrotechnical Commission, IEC 61508, Functional safety of electrical/electronic/programmable electronic safety-related systems, Edition 2.0, Apr 2010.

[ISO26262] International Organization for Standardization. ISO 26262 Road Vehicles –Functional safety–. 1st Edition, 2011-11-15.

[Model] The qualification model for the Testwell CTC++. It is contained in the qualification kit and can be opened and changed using the [TCA].

[SAFECOMP12] Determining Potential Errors in Tool Chains: Strategies to Reach Tool Confidence According to ISO 26262, SAFECOMP 2012, Wildmoser, Philipps, Slotosch

[TAG] Tool Application Guide (Tool Safety Manual) for Testwell CTC++

[TAU_UG] Tool Automation Unit for Testwell CTC++, contained in this qualification kit contained in the documentation of this kit in the file: TAU_UserGuide.docx

[TCA] Tool Chain Analyzer, tool available on www.validas.de/TCA.html Version 1.10.0

[TCA_UM] Tool Chain Analyzer, Version 1.10.0, User Manual, (<TCAHome>/plugins/Documentation/UserManual.pdf)

[TCR] Tool Classification Report for Testwell CTC++

[TQP] Tool Qualification Plan for Testwell CTC++

[TQR] Tool Qualification Report for Testwell CTC++

8 Appendix: Requirements Tracing to Safety Standards

The relevant safety standards have comparable approaches to tool qualification. In all standards the goal is to ensure that the tools can not impact the safety of the product, i.e. that all potential errors of the tool are either absent or cannot impact the safety. And all standards do this by a combination of application and installation methods. The application methods are safety guidelines that explain how to use the tool and avoid/detect the potential errors, while the installation methods ensure that the installed tool works as expected, e.g. by testing it to show the absence of the potential errors.

All standards have a classification phase to determine the required confidence into the tool and a qualification phase that provides this confidence or restricts the usage of the tools to confident scenarios. However the classification and qualification methods differ in some details. Nevertheless our qualification approach is suitable for all standards and does not require unnecessary work.

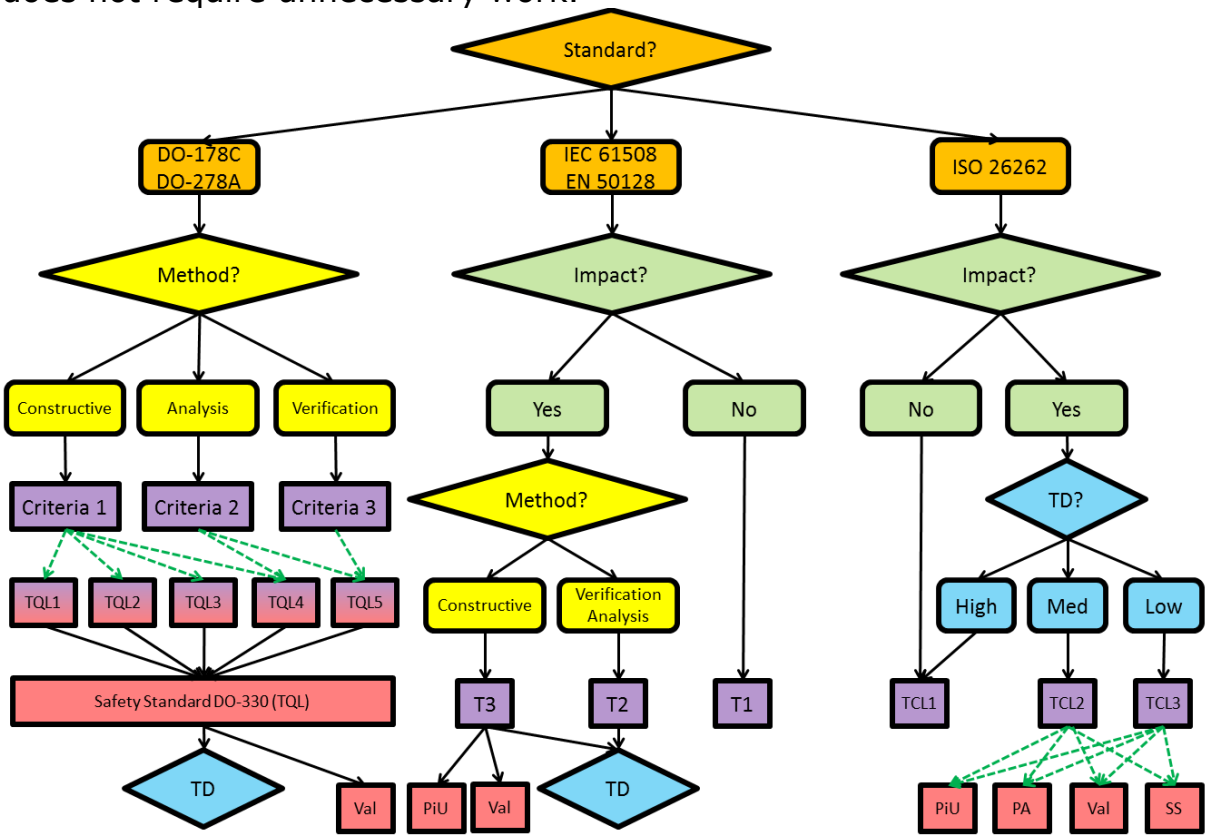


Figure 36: Comparison of Qualification Approaches

Figure 36 gives an overview on the different approaches. The main difference between ISO 26262 and the other standards is that the classification of tools depends on the analysis of the potential errors and their detection, which increases the variability of the classification. The impact and the supported methods/processes are considered in all standards as part of the classification. While the ISO does not differentiate between kinds of the tools the other standards do and classify the tools for constructive methods (e.g. code generators and compilers) as more critical than the other tools verification, automation and analytic tools. The

results of the classification is the confidence needs (represented in pink color in Figure 36). The DO expresses the tool confidence requirement by criteria 1-3 the ISO 26262 as tool confidence levels and IEC 61508 and EN 50128 as tool classes T1-T3. The next step is to derive the qualification methods from the qualification needs of the tool and the criticality of the developed software. ISO 26262 and DO 178C, DO 278A do have tables that map the software criticality to qualification methods, e.g. a validation is required from ISO 26262 for TQL 3 tool in ASIL C and D projects. In DO the qualification methods are determined by the tool qualification level (TQL) that is the interface to the DO-330 and determines the development of the tool, which is a specific qualification method. This criticality dependent selection of qualification methods is depicted in Figure 36 using green dotted lines. The qualification methods differ also. While the DO allows only the development according to the DO-330, a safety standard (SS), the other standards include also a proven in use argumentation (PiU) and a process assessment (PA). Since DO-330 requires also a validation, the validation is the only method that is applicable in every standard. Furthermore the analysis of potential tool errors and their detection (TD) is required in every approach for tools that have impact.
Therefore this classification report contains the determination of the tool confidence need and the analysis of the potential errors and their detection, that belongs to the classification in the ISO 26262.
The qualification method is tool validation by testing the safety relevant parts of the tool. The safety relevant parts are determined by the tool chain analysis together with the determination of the qualification need.

We formalize the tool chain to determine the required confidence using the following model:

- Use case: describes an application scenario of the tool
- Feature: a tool function utilized in use cases
- Potential error: a potential error that could occur during the application of a tool
- Error mitigation: a check or restriction applied during the tool operation phase
- Qualification: a method to show that a tool or a feature satisfies its specified requirements by demonstrating the absence of potential errors.

8.1 Requirements of ISO 26262

The requirements for tools in the ISO 26262 are distributed in several parts. Figure 37 shows the relations between the ISO 26262 requirements for tool qualification[5].

[5] Note that the evaluation requirements are more relevant for the tool evaluation report [TCR] and therefore here only referred as far as they are relevant for the safety manual. Also the qualification requirements are only listed as far as relevant for this safety guide.

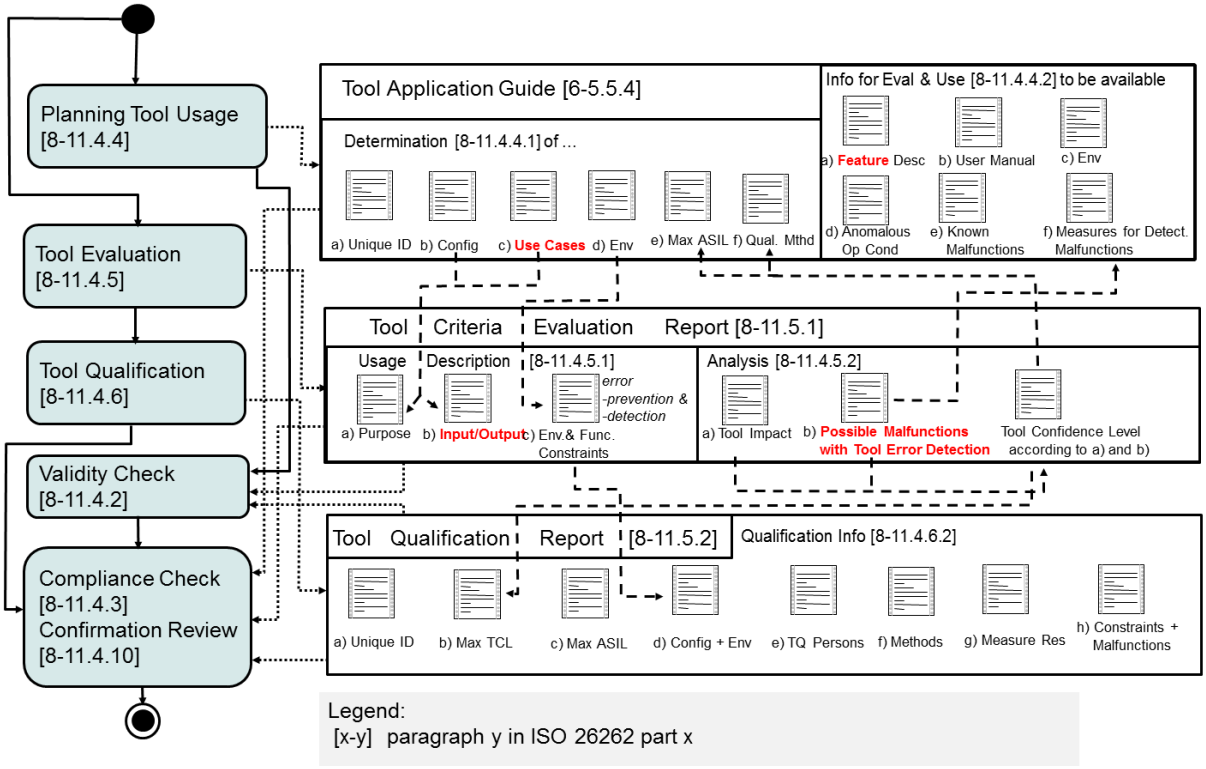


Figure 37: ISO 26262 Tool Qualification Requirements

The following requirements are stated in the ISO 26262:

- Part 6. Chapter 5: Initiation of product development at the software level
  - [ISO_6_5_45] For each sub-phase of software development select methods and tools including guidelines for their application.
  - [ISO_6_5_46a] unambiguous definition.
  - [ISO_6_5_54] **Tool application guidelines** resulting from requirements 5.4.5 and 5.4.6.
- Part 8. Chapter 11: Confidence in the use of software tools contains the analysis and the qualification methods
  - [ISO_8_11_42] Validity of predetermined tool confidence level or qualification
  - [ISO_8_11_421] If the confidence level evaluation or qualification of a software tool is performed independently from the development of a particular safety-related item or element, the validity of this predetermined tool confidence level or qualification shall be confirmed, in accordance with ISO 26262-2:2011, Table 1, prior to the software tool being used for the development of a particular safety-related item or element
  - [ISO_8_11_431] When using a software tool, it shall be ensured that its usage, its determined environmental and functional constraints and its general operating conditions comply with its evaluation criteria or its qualification.

- [ISO_8_11_441] The usage of a software tool shall be planned, including the determination of:
  - [ISO_8_11_441a] the identification and version number of the software tool,
  - [ISO_8_11_441b] the configuration of the software tool,
  - [ISO_8_11_441c] the use cases of the software tool,
  - [ISO_8_11_441d] the environment in which the software tool is executed,
  - [ISO_8_11_441e] the maximum ASIL of all the safety requirements, allocated to the item or the element that can be violated, if the software tool is malfunctioning and producing corresponding erroneous output, and
  - [ISO_8_11_441f] the measures for the detection of malfunctions and the corresponding erroneous output of the software tool identified during the determination of the required level of confidence for this software tool.
- [ISO_8_11_442] To ensure the proper evaluation or usage of the software tool, the following information shall be available:
  - [ISO_8_11_442a] description of the features, functions and technical properties of the software tool,
  - [ISO_8_11_442b] the user manual or other usage guides, if applicable,
  - [ISO_8_11_442c] a description of the environment required for its operation,
  - [ISO_8_11_442d] a description of the expected behaviour of the software tool under anomalous operating conditions, if applicable,
  - [ISO_8_11_442e] a description of known software tool malfunctions and the appropriate safeguards, avoidance or work-around measures, if applicable, and
  - [ISO_8_11_442f] the measures for the detection of malfunctions and the corresponding erroneous output of the software tool identified during the determination of the required level of confidence for this software tool.
- [ISO_8_11_451] The description of the usage of a software tool shall contain the following information:
  - [ISO_8_11_451a] the intended purpose
  - [ISO_8_11_451b] the inputs and expected outputs, and
  - [ISO_8_11_451c] the environmental and functional constraints, if applicable.
- [ISO_8_11_452] The intended usage of the software tool shall be analyzed and evaluated to determine:
  - [ISO_8_11_452a] the possibility that a malfunction of a particular software tool can introduce or fail to detect errors in a safety-related item or element being developed. This is expressed by the classes of Tool Impact

        - [ISO_8_11_452b] the confidence in measures that prevent the software tool from malfunctioning and producing corresponding erroneous output, or in measures that detect that the software tool has malfunctioned and has produced corresponding erroneous output. This is expressed by the classes of Tool error Detection (TD):
            - TD1 shall be selected if there is a high degree of confidence that a malfunction and its corresponding erroneous output will be prevented or detected;
            - TD2 shall be selected if there is a medium degree of confidence that a malfunction and its corresponding erroneous output will be prevented or detected;
            - TD3 shall be selected in all other cases.
    - [ISO_8_11_462] The qualification of the software tool shall be documented including the following:
        - [ISO_8_11_462a] the unique identification and version number of the software tool
        - [ISO_8_11_462b] the maximum Tool Confidence Level for which the software tool is classified together with a reference to its evaluation analysis,
        - [ISO_8_11_462c] the pre-determined maximum ASIL, or specific ASIL, of any safety requirement which might be violated if the software tool is malfunctioning and produces corresponding erroneous output
        - [ISO_8_11_462d] the configuration and environment for which the software tool is qualified,
        - [ISO_8_11_462e] the person or organization who carried out the qualification,
        - [ISO_8_11_462f] the methods applied for its qualification in accordance with 11.4.6.1
        - [ISO_8_11_462g] the results of the measures applied to qualify the software tool, and
        - [ISO_8_11_462h] the usage constraints and malfunctions identified during the qualification, if applicable.
    - [ISO_8_11_410] Confirmation review of qualification of a software tool
      This subclause applies to ASILs (B), C, D, in accordance with 4.3. The confidence in the use of the software tool shall be evaluated in accordance with ISO 26262-2:2011 Table 1 to ensure:
        - [ISO_8_11_410a] the correct evaluation of the required level of confidence in the software tool, and

        - [ISO_8_11_410b] the appropriate qualification of the software tool in accordance with its required level of confidence.

8.2 Requirements of IEC 61508

The requirements for tools in the IEC 61508 are distributed in several parts. Part 4 contains the relevant tool definitions for the classes T2 (test tools) and T3 (constructive tools), while Part 3 contains software requirements.

- Part 3: Software requirements
    - [IEC_3_7443] The selection of the off-line support tools shall be justified
    - [IEC_3_7444] All off-line support tools in classes T2 and T3 shall have a specification or product documentation which clearly defines the behavior of the tool and any instructions or constraints on its use.
    - [IEC_3_7445] An assessment shall be carried out for offline support tools in classes T2 and T3 to determine the level of reliance placed on the tools, and the potential failure mechanisms of the tools that may affect the executable software. Where such failure mechanisms are identified, appropriate mitigation measures shall be taken.
    - [IEC_3_7446] For each tool in class T3, evidence shall be available that the tool conforms to its specification or documentation. Evidence may be based on a suitable combination of history of successful use in similar environments and for similar applications (within the organization or other organizations), and of tool validation as specified in 7.4.4.7
    - [IEC_3_7447] The results of tool validation shall be documented covering the following results:
        - [IEC_3_7447a] a chronological record of the validation activities;
        - [IEC_3_7447b] the version of the tool product manual being used;
        - [IEC_3_7447c] the tool functions being validated;
        - [IEC_3_7447d] tools and equipment used;
        - [IEC_3_7447e] the results of the validation activity; the documented results of validation shall state either show that the software has passed the validation or the reasons for its failure;
        - [IEC_3_7447f] test cases and their results for subsequent analysis;
        - [IEC_3_7447g] discrepancies between expected and actual results.
    - [IEC_3_7448] Where the conformance evidence of 7.4.4.6 is unavailable, there shall be effective measures to control

failures of the executable safety related system that result from faults that are attributable to the tool.
    - [IEC_3_7449] The compatibility of the tools of an integrated toolset shall be verified.
    - [IEC_3_74415] Where off-line support tools of classes T2 and T3 generate items in the configuration baseline, configuration management shall ensure that information on the tools is recorded in the configuration baseline. This includes in particular:
        - [IEC_3_74415a] the identification of the tool and its version;
        - [IEC_3_74415b] the identification of the configuration baseline items for which the tool version has been used;
        - [IEC_3_74415c] the way the tool was used (including the tool parameters, options and scripts selected) for each configuration baseline item.
    - [IEC_3_74416] Configuration management shall ensure that for tools in classes T2 and T3, only qualified versions are used.
    - [IEC_3_74417] Configuration management shall ensure that only tools compatible with each other and with the safety-related system are used.
    - [IEC_3_74418] Each new version of off-line support tool shall be qualified. This qualification may rely on evidence provided for an earlier version if sufficient evidence is provided that:
    - [IEC_3_74418a] the functional differences (if any) will not affect tool compatibility with the rest of the toolset; and
    - [IEC_3_74418b] the new version is unlikely to contain significant new, unknown faults.
- Part 4: Definitions and abbreviations
    - [IEC_4_3211] software off-line support tool: software tool that supports a phase of the software development lifecycle and that cannot directly influence the safety-related system during its run time. Software off-line tools may be divided into the following classes:
        - [IEC_4_3211a] T1 generates no outputs which can directly or indirectly contribute to the executable code (including data) of the safety related system.
        - [IEC_4_3211b] T2 supports the test or verification of the design or executable code, where errors in the tool can fail to reveal defects but cannot directly create errors in the executable software;
        - [IEC_4_3211c] T3 generates outputs which can directly or indirectly contribute to the executable code of the safety related system.

8.3 Requirements of EN 50128

In Section 7.4.4 (“Support tools and languages”) of [EN50128] the following requirements are described. Many of them are covered by the

tool qualification (see Section) and have a VS-EN-ID. The others are argued to be not applicable for this qualification and shall be covered by the surrounding safety process.

Req ID (if applicable)	Source	Requirement Text
NA, since Software tool planning is out of scope of this report	6.7.4.1	Software tools shall be selected as a coherent part of the software development activities
See [TCR]	6.7.4.2	The selection of the tools in classes T2 and T3 shall be justified (see 7.3.4.12). The justification shall include the identification of potential failures which can be injected into the tools output and the measures to avoid or handle such failures.
See [TAG]	6.7.4.3	All tools in classes T2 and T3 shall have a specification or manual which clearly defines the behaviour of the tool and any instructions or constraints on its use.
See sub-items	6.7.4.4	For each tool in class T3, evidence shall be available that the output of the tool conforms to the specification of the output or failures in the output are detected. Evidence may be based on the same steps necessary for a manual process as a replacement for the tool and an argument presented if these steps are replaced by alternatives (e. g. validation of the tool). Evidence may also be based on
NA, this method is not used in this qualification	6.7.4.4.a	a suitable combination of history of successful use in similar environments and for similar applications (within the organization or other organizations),
See VS-EN-11 to -17	6.7.4.4.b	tool validation as specified in 6.7.4.5
See [TCR] and [TAG]	6.7.4.4.c	diverse redundant code which allows the detection and control of failures resulting in faults introduced by a tool,
See [TCR]	6.7.4.4.d	compliance with the safety integrity levels derived from the risk analysis of the process and procedures including the tools
See [TCR]	6.7.4.4.e	other appropriate methods for avoiding or handling failures introduced by tools
See sub-items	6.7.4.5	The results of tool validation shall be documented covering the following results
VS-EN-11	6.7.4.5.a	a record of the validation activities
VS-EN-12	6.7.4.5.b	the version of the tool manual being used
VS-EN-13	6.7.4.5.c	the tool functions being validated
VS-EN-14	6.7.4.5.d	tools and equipment used
VS-EN-15	6.7.4.5.e	the results of the validation activity; the documented results of validation shall state either that the software has passed the validation or the reasons for its failure
VS-EN-16	6.7.4.5.f	test cases and their results for subsequent analysis
VS-EN-17	6.7.4.5.g	discrepancies between expected and actual results
VS-EN-18	6.7.4.6	Where the conformance evidence of 6.7.4.4 is unavailable, there shall be effective measures to

		control failures of the executable safety related software that result from faults that are attributable to the tool
See sub-items	6.7.4.7	The software or design representation (including a programming language) selected shall
VS-EN-20	6.7.4.7.a	have a translator which has been evaluated for fitness for purpose including, where appropriate, evaluated against the international or national standards
VS-EN-21	6.7.4.7.b	match the characteristics of the application
See [TAG]	6.7.4.7.c	contain features that facilitate the detection of design or programming errors
VS-EN-23	6.7.4.7.d	support features that match the design method.
NA this is just an explanation	6.7.4.7.1	A programming language is one of a class of representations of software or design. A Translator converts a software or design representation (e.g. text or a diagram) from one abstraction level to another level. Examples of Translators include: design refinement tools, compilers, assemblers, linkers, binders, loaders and code generation tools.
VS-EN-25	6.7.4.7.2	The evaluation of a Translator may be performed for a specific application project, or for a class of applications. In the latter case all necessary information on the tool regarding the intended and appropriate use of the tool shall be available to the user of the tool. The evaluation of the tool for a specific project may then be reduced to checking general suitability of the tool for the project and compliance to the "specification or manual" (i.e. proper use of the tool). Proper use might include additional verification activities within the specific project.
VS-EN-26	6.7.4.7.3	A validation suite may be used to evaluate the fitness for purpose of a Translator according to defined criteria, which shall include functional and non-functional requirements. For the functional Translator requirements, dynamic testing may be a main validation technique. If possible an automatic testing suite shall be used.
Software selection is not in the scope of tool qualification	6.7.4.8	Where 6.7.4.7 cannot be fully satisfied, the fitness for purpose of the language, and any additional measures which address any identified shortcomings of the language shall be justified and evaluated.
See [TCR]	6.7.4.9	Where automatic code generation or similar automatic translation takes place, the suitability of the automatic Translator for safety-related software development shall be evaluated at the point in the development lifecycle where development support tools are selected
Configuration Management is not in the scope of tool qualification	6.7.4.10	Configuration management shall ensure that for tools in classes T2 and T3, only justified versions are used.
No support for delta qualifications (classification in [TCR] may be reused)	6.7.4.11	Each new version of a tool that is used shall be justified (see Table 1 in [EN50128]). This justification may rely on evidence provided for an earlier version if sufficient evidence is provided that

VS-EN-31	6.7.4.12	The relation between the tool classes and the applicable sub-clauses is defined within Table 1 in [EN50128].

Table 1 Validation Suite Requirements from EN 50128

8.4 Requirements of DO-330 (Operational Parts)

The DO-330 is safety standard for the development of tools and therefore only the parts for the determination of the required confidence (classification) and the tool operational requirements are considered. The used notions of Tool operational requirements (TORs) and Tool Requirements (TRs) correspond to our terms use case and feature.

- [DO_330_41] Determining the Tool Qualification Needs: During the software planning process:
  - [DO_330_41a] the tools used in the scope of the software life cycle process are identified
  - [DO_330_41b] each tool's intended use is described
  - [DO_330_41c] the need for tool qualification is defined
  - [DO_330_41d] the TQLs are determined
  - [DO_330_41e] the tool qualification stakeholders and their assigned roles and responsibilities are identified
  - [DO_330_41f] the tool operational environment is described. If there are multiple tool operational environments, then all such environments are described.
- [DO_330_1011]: Tool-Specific Information in PSAC: The PSAC should include the following information about the need for tool qualification
  - [DO_330_1011a] Identification of the tool and its intended use, including its impact in the software life cycle process
  - [DO_330_1011b] Details of the certification credit sought through tool use for eliminating, reducing, or automating the process(es)
  - [DO_330_1011c] Substantiation of the maturity and technical background of any technology or theory (for example, mathematical theory) implemented in the tool to show its applicability
  - [DO_330_1011d] The TQL proposed for the tool and supporting justification
  - [DO_330_1011e] Tool source (for example, in-house, COTS, third party)
  - [DO_330_1011f] The stakeholders involved in the tool qualification and their specific roles and responsibilities, including who is responsible for satisfying specific objectives
  - [DO_330_1011g] Description of the Tool Operational Requirements definition process (see 5.1), tool operation integration process (see 5.3), and tool operational V&V process (see 6.2) (or a reference to where these processes will be described)
  - [DO_330_1011h] Description of the tool operational

environment in which the tool will be used

- [DO_330_1031]: Tool Operational Requirements: 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:
    - [DO_330_1031a] Description of the context of the tool use, including interfaces with other tools and integration of the tool output files into the resultant software
    - [DO_330_1031b] Description of the tool operational environment(s) (where the tool will be installed).
    - [DO_330_1031c] Description of input files, including format, language definition, etc.
    - [DO_330_1031d] Description of output files, including format and contents
    - [DO_330_1031e] Requirements for all the tool functions and technical features used to satisfy the identified software life cycle process(es).
    - [DO_330_1031f] 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.
    - [DO_330_1031g] 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.
    - [DO_330_1031h] Description of the operational use of the tool (including selected options, parameters values, command line, etc.).
    - [DO_330_1031i] Performance requirements specifying the behavior of the tool output.
- [DO_330_1034] Tool Operational Verification and Validation Results. The tool operational verification and validation process produces the Tool Operational Verification and Validation Results, which should
    - [DO_330_1034a] For each review, analysis, and test, indicate each procedure that passed or failed during the activities and the final pass/fail results.
    - [DO_330_1034b] Identify the configuration item or tool version reviewed, analyzed, or tested
    - [DO_330_1034c] Include the results of tests, reviews, and analyses
    - [DO_330_1034d] Record and track any discrepancies found using the problem reporting process
- [DO_330_D2] To reduce a tool's qualification level, the reduction needs to be justified by performing a tool use and impact analysis. This analysis needs to evaluate the overall use of the tool in the development process and its impact on the software being produced.

8.5 Satisfaction of ISO 26262 Requirements

The identified requirements from Section 8.1 are satisfied as follows:

- [ISO_6_5_45]
  this document contains an unique identification of the Testwell CTC++ (Section 5.1) and the guidelines how to use it
- [ISO_6_5_46a]
  see the definition of the Testwell CTC++ in Section 5.1 in [TAG]
- [ISO_6_5_54]
  see the guidelines how to apply the tool in Sections 5, 6,7 in [TAG]
- [ISO_8_11_42]
  see TG_TCA_52_2 in Section 5.1 in [TAG]
- [ISO_8_11_421]
  see TG_TCA_52_2 in Section 5.1 in [TAG] (the TCL has been determined in [TCR])
- [ISO_8_11_431]
  see Sections 5, 6, and 7 in [TAG]
- [ISO_8_11_441]
  see tracing of [ISO_8_11_441a] to [ISO_8_11_441f]
- [ISO_8_11_441a]
  see Section 5.1 in [TAG]
- [ISO_8_11_441b]
  see Section 5.1 in [TAG]
- [ISO_8_11_441c]
  see Section 6 in [TAG]
- [ISO_8_11_441d]
  see Section 5.1 in [TAG]
- [ISO_8_11_441e]
  see TG_TCA_52_4 in Section 5.1 in [TAG]
- [ISO_8_11_441f]
  see the results in [TCR]
- [ISO_8_11_442]
  see tracing of [ISO_8_11_442a] to [ISO_8_11_442f]
- [ISO_8_11_442a]
  see [TCR] and Section 5.3 in [TAG]
- [ISO_8_11_442b]
  see Section 5.1 in [TAG]
- [ISO_8_11_442c]
  see Section 5.1 in [TAG]
- [ISO_8_11_442d]
  see Section 7.3 in [TAG]
- [ISO_8_11_442e]
  see Section 5.4.2 in [TAG]
- [ISO_8_11_442f]
  see Sections 7.1 and 7.2 in [TAG]
- [ISO_8_11_451]
  see tracing of [ISO_8_11_451a] to [ISO_8_11_451c]
- [ISO_8_11_451a]
  see the description of use cases in Section 6 in [TAG]

- [ISO_8_11_451b]
  see the description of features and use cases in Sections 5 and 6 in [TAG]
- [ISO_8_11_451c]
  see guidelines in Section 5.1 and OG_TCA_7_4 in Section 7 in [TAG]
- [ISO_8_11_452], [ISO_8_11_452a], [ISO_8_11_452b]
  has been determined in [TCR]
- [ISO_8_11_462]
  see tracing of [ISO_8_11_462a] to [ISO_8_11_462h]
- [ISO_8_11_462a]
  see Section 5.1 in [TAG]
- [ISO_8_11_462b]
  see TG_TCA_52_2 and in [TCR]
- [ISO_8_11_462c]
  see TG_TCA_52_4 in Section 5.2 in [TAG]
- [ISO_8_11_462d]
  see TG_TCA_52_3 in Section 5.2, TG_TCA_53_2 in Section 5.3, and UG_TCA_61_3 in Section 6.1 in [TAG]
- [ISO_8_11_462e]
  see TG_TCA_52_3 in Section 5.2, TG_TCA_53_2 in Section 5.3, and UG_TCA_61_3 in Section 6.1 in [TAG]
- [ISO_8_11_462f]
  see TG_TCA_52_3 in Section 5.2, TG_TCA_53_2 in Section 5.3, and UG_TCA_61_3 in Section 6.1 in [TAG]
- [ISO_8_11_462g]
  see TG_TCA_54_5 in Section 5.4 in [TAG]
- [ISO_8_11_462h]
  see TG_TCA_7_5 in Section 7 in [TAG]
- [ISO_8_11_410]
  see [ISO_8_11_410a] and [ISO_8_11_410b]
- [ISO_8_11_410a]
  see TG_TCA_52_1 in Section 5.2 in [TAG]
- [ISO_8_11_410b]
  see TG_TCA_52_2 in Section 5.2 in [TAG]

8.6 Satisfaction of IEC-61508 Requirements

The identified requirements from section 8.2 are satisfied as follows:

- [IEC_3_7443] is done in [TCR]
- [IEC_3_7444] qualification needs are satisfied by qualification, which is verified in Section 5.2 in [TAG]
- [IEC_3_7445] is done for all off-line tools [TCR].
- [IEC_3_7446] tool qualification provides the evidence and is checked in Section 5.2 in [TAG]
- [IEC_3_7447] see [IEC_3_7447a] to [IEC_3_7447g]
- [IEC_3_7447a] see qualification report of the tool and the checks in TG_TCA_52_3, TG_TCA_53_2 and UG_TCA_61_3

- [IEC_3_7447b] see Section 5.1;
- [IEC_3_7447c] see Section 5.3;
- [IEC_3_7447d] see tool qualification plan;
- [IEC_3_7447e] see qualification report
- [IEC_3_7447f] part of qualification report
- [IEC_3_7447g] part of qualification report
- [IEC_3_7448] proposed in tools [TCR] and validated by confirmation review in TG_TCA_52_3, TG_TCA_53_2 and UG_TCA_61_3
- [IEC_3_7449] is analyzed by the artifact flow in [TCR]
- [IEC_3_74415] satisfied by [IEC_3_74415a] to [IEC_3_74415c]
- [IEC_3_74415a] see Section 5.1;
- [IEC_3_74415b] see TG_TCA_51_3;
- [IEC_3_74415c] see use case description in Section 6.1 and UG_TCA_61_1.
- [IEC_3_74416] see TG_TCA_52_5, other tools are TCL 1.
- [IEC_3_74417] is analyzed by the artifact flow in [TCR]
- [IEC_3_74418] is implied by TG_TCA_52_5:
- [IEC_3_74418a] is implied by TG_TCA_52_5 and TG_TCA_7_5
- [IEC_3_74418b] reuse qualification tests in tool qualification plan.
- [IEC_4_3211] definitions (see [IEC_4_3211a] until [IEC_4_3211b])
- [IEC_4_3211a] T1 has no impact and is therefore TCL1
- [IEC_4_3211b] T2 can oversee errors and is therefore TCL2 or TCL3;
- [IEC_4_3211c] T3 can introduce errors and is therefore TCL2 or TCL3.

8.7 Satisfaction of EN 50128

The requirements from section 8.3 for T3 tools for are covered as follows:

1. VS-EN-11: The tool qualification report which extends this tool qualification plan documents the performed activities
2. VS-EN-12: The version of the tool is contained in Section 5 of [TQP] and the [TAG].
3. VS-EN-13: The validated tool functions are modeled as Features and are listed in Section 5 of [TQP]
4. VS-EN-14: The TAU is described in [TAU_UG] that is referred in this qualification plan and the resulting qualification report.
5. VS-EN-15: The tool qualification report which extends this tool qualification plan documents the result of the validation
6. VS-EN-16: The test report which is generated from the TAU by processing the test plan contains the test results.
7. VS-EN-17: The test report contains also the discrepancies (failed tests)
8. VS-EN-18: [TCR] and [TAG] contain effective measures for features that are not qualified according to this qualification plan.
9. VS-EN-20: The qualification plan for the Testwell CTC++ [TQP] satisfies international standards

10. VS-EN-21: The matching to the application is achieved by selecting the required features of the tool and verified by comparing the code coverage between the qualification and the application of the Testwell CTC++. This is documented in [TCR] and [TAG].
11. VS-EN-23: The required features have been selected from the user to match is development process and are listed in Section 5 of [TQP].
12. VS-EN-25: The qualification in [TQP] is project specific by selecting the features and comparing the code coverage in the Testwell CTC++ during qualification with the application.
13. VS-EN-26: The selected qualification method is validation of the selected features and contains functional tests (see Section 6.5.2 of [TQP]) as well as robustness tests (see Section 6.5.3 of [TQP])
14. VS-EN-31: The tool qualification plan [TQP] captures all relevant requirements of the EN50128 for T3 tools. For T1 and T2 tools no qualification is required, but an analysis of potential errors as was done during the creation of the qualification kit. This is configured from the user during application of the kit and documents the relevant analysis in [TCR] and [TAG] that justify the selection of the Testwell CTC++.

8.8 Satisfaction of Requirements of DO-330 (Operational Parts)

The identified requirements from section 8.3 are satisfied as follows:

- [DO_330_41] see [DO_330_41a] until [DO_330_41f]
- [DO_330_41a] is done in [TCR]
- [DO_330_41b] see use cases in Section 6.1
- [DO_330_41c] is done in [TCR]
- [DO_330_41d] is done in [TCR] and can be mapped to TQLs
- [DO_330_41e] is done in tool qualification plan
- [DO_330_41f] see section 5.1
- [DO_330_1011]: see [DO_330_1011a] to [DO_330_1011h]
- [DO_330_1011a] see section 5.1 and 6.1
- [DO_330_1011b] is done in [TCR]
- [DO_330_1011c] is done in tool qualification plan
- [DO_330_1011d is done in [TCR] and can be mapped to TQLs
- [DO_330_1011e] see section 5.1
- [DO_330_1011f] is done in tool qualification plan
- [DO_330_1011g] see section 7
- [DO_330_1011h] see section 5.1 and 6.1
- [DO_330_1031] see [DO_330_1031a] to [DO_330_1031i]
- [DO_330_1031a] is done in [TCR] and section 5.1
- [DO_330_1031b] see section 5.1
- [DO_330_1031c] is done in [TCR]
- [DO_330_1031d] is done in [TCR]

- [DO_330_1031e] is done in [TCR] and sections 5.3 and 6.1
- [DO_330_1031f] is done in tool qualification plan.
- [DO_330_1031g] see section 5.1
- [DO_330_1031h] see use-cases in section 6.1
- [DO_330_1031i] Performance is not considered for safety relevant failures.
- [DO_330_1034] see [DO_330_1034a] until [DO_330_1034d]
    - [DO_330_1034a] is done in tool qualification plan.
    - [DO_330_1034b] is done in tool qualification plan.
    - [DO_330_1034c] is done in tool qualification report
      [DO_330_1034d] is done in tool qualification report
- [DO_330_D2] is done in [TCR].