Oscar Slotosch, Validas AG

Enabling Development of Qualifiable Eclipse-based Tools: Vision and Concept

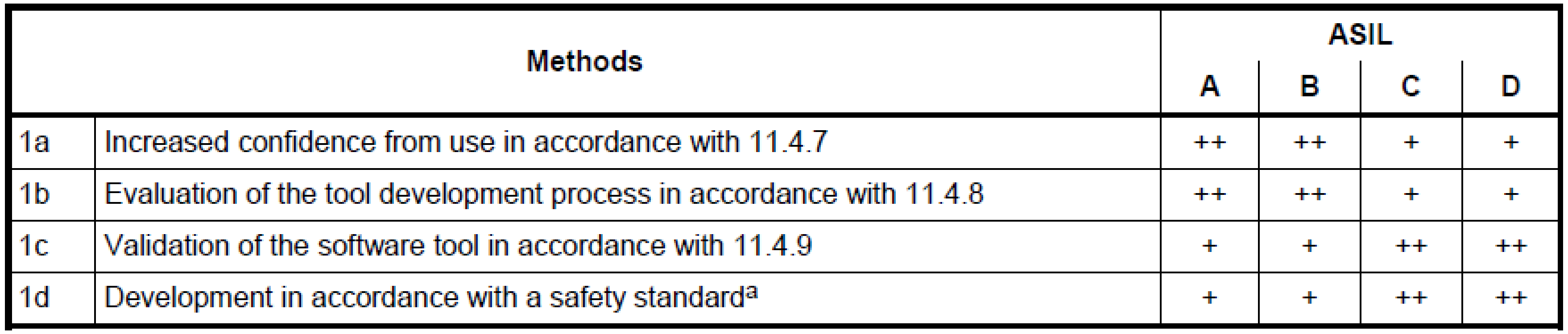
Content

- **Tool Qualification Requirements from Standards**
- **Tool Qualification Roadmap**
  - Vision
  - DO-330
  - Concept
    - Model-based Tool Qualification
    - Examples
      - Processes
      - Documents
    - Status: May 2012
- **Current Demonstration Examples**
- **Summary**

Tool Qualification (Summary)

- **Standards require tool qualification: ISO 26262, IEC 61508, DO, EN 50128**
- **Qualification process:**
  - Classify all used tools (Impact, Use-Cases, Artifacts)
  - Qualify critical tools
  - Use tools
- **Qualification Methods ISO 26262**

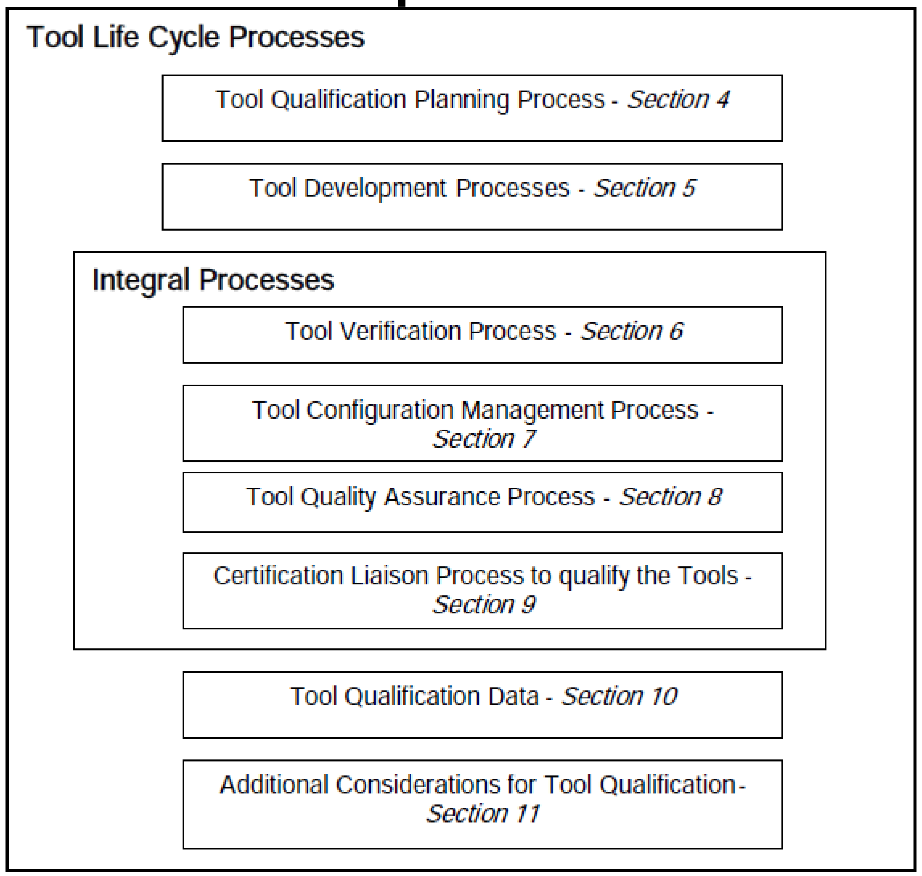
Table 4 — Qualification of software tools classified TCL3



Methods		ASIL			
		A	B	C	D
1a	Increased confidence from use in accordance with 11.4.7	++	++	+	+
1b	Evaluation of the tool development process in accordance with 11.4.8	++	++	+	+
1c	Validation of the software tool in accordance with 11.4.9	+	+	++	++
1d	Development in accordance with a safety standard[a]	+	+	++	++

- **Qualification Method DO-330 Development in accordance with a safety standard:**
  - Processes Requirements
  - Required Documents
  - Required Verification
  - Required Qualification Process
  - …

DO-330 Required Processes



Tool Life Cycle Processes

Tool Qualification Planning Process - *Section 4*

Tool Development Processes - *Section 5*

Integral Processes

Tool Verification Process - *Section 6*

Tool Configuration Management Process - *Section 7*

Tool Quality Assurance Process - *Section 8*

Certification Liaison Process to qualify the Tools - *Section 9*

Tool Qualification Data - *Section 10*

Additional Considerations for Tool Qualification - *Section 11*

Content

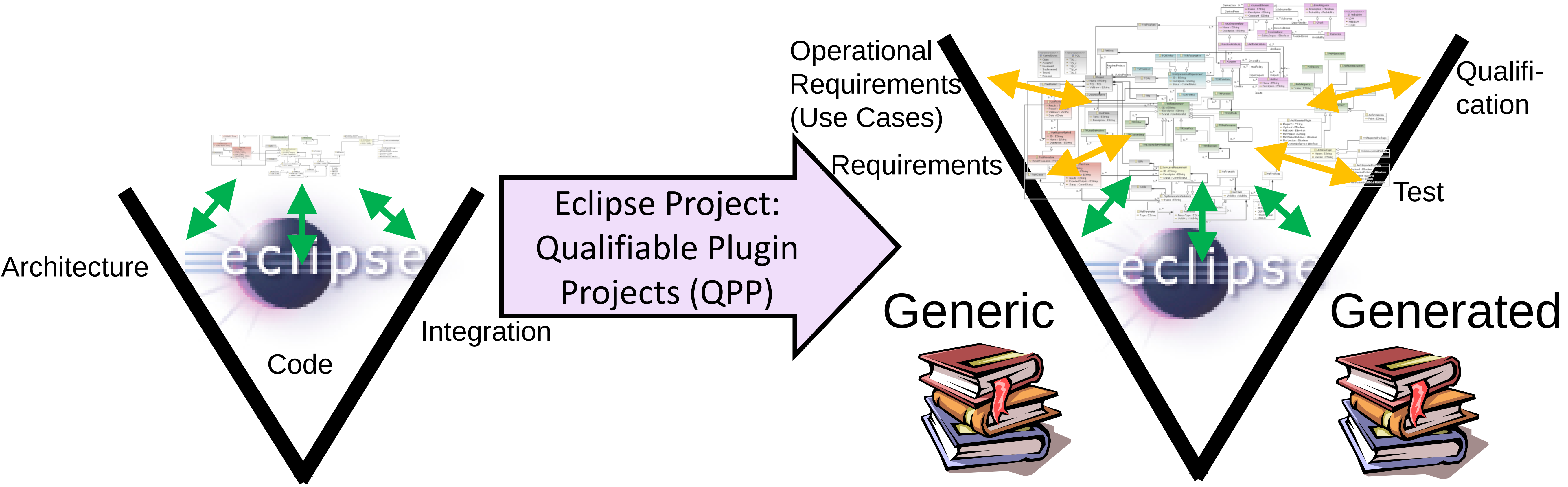
- **Tool Qualification Requirements from Standards**
- **Tool Qualification Roadmap**
  - Vision
  - DO-330
  - Concept
    - Model-based Tool Qualification
    - Examples
      - Processes
      - Documents
    - Status: May 2012
- **Current Demonstration Examples**
- **Summary**

Vision: Eclipse Development Process

- **Currently Eclipse does not support qualification**
- **There is a road towards tool qualification for Eclipse, see http://wiki.eclipse.org/Auto_IWG_WP5**
- **DO-330 is a safety standard for tools**

Current Process

New Extended Process



Architecture

Code

Integration

Eclipse Project: Qualifiable Plugin Projects (QPP)

Operational Requirements (Use Cases)

Requirements

Qualifi-cation

Test

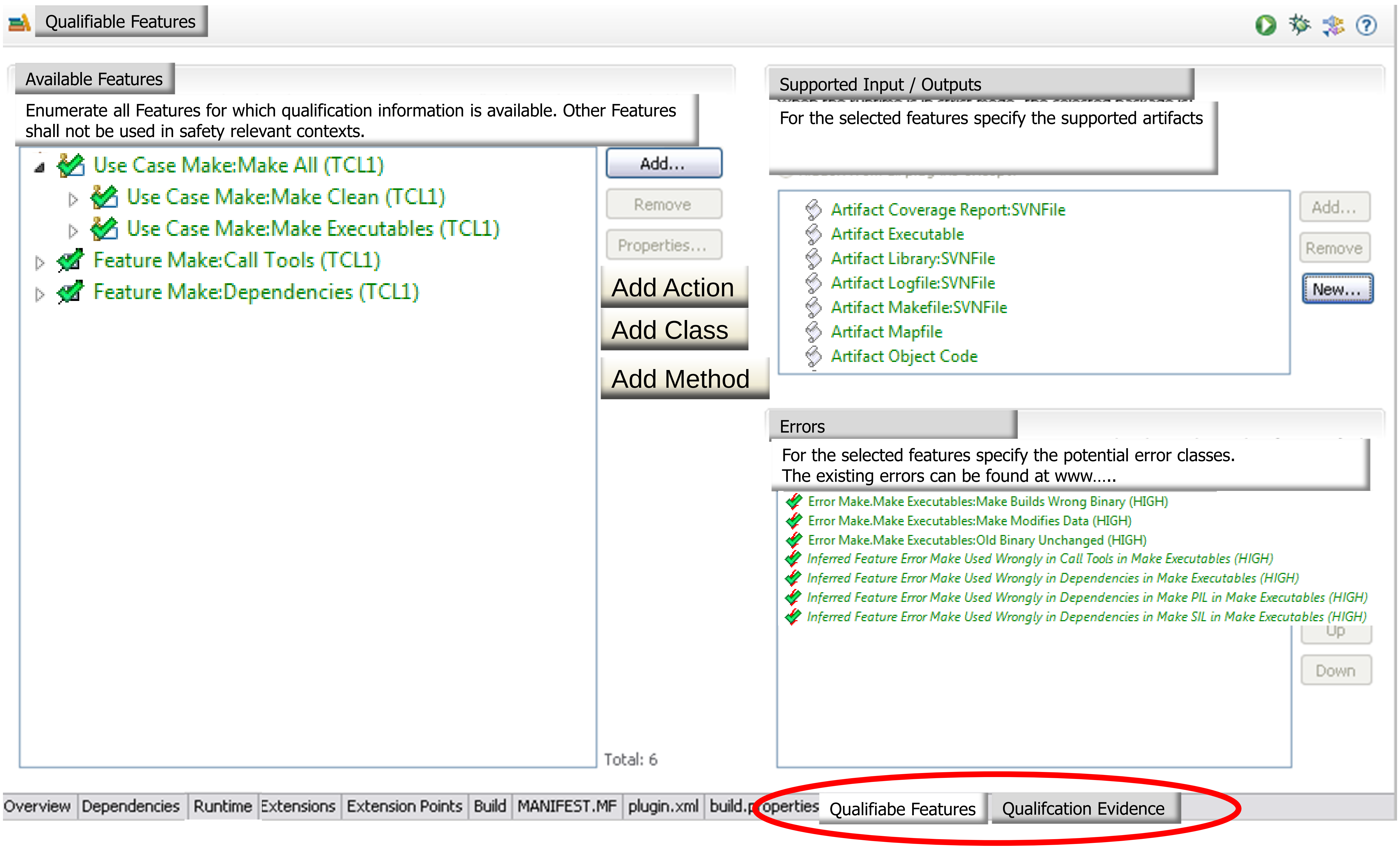
Generic

Generated

How-To Qualify Tools
according DO-330
Tool Development Plan
Tool Verification Plan
…

Requirements-Specification
Design-Specification
Test-Specification
Tool Analysis (TCL/PSAC)
…

Vision: Eclipse Classification Data



Qualifiable Features

Available Features

Enumerate all Features for which qualification information is available. Other Features shall not be used in safety relevant contexts.

- Use Case Make:Make All (TCL1)
  - Use Case Make:Make Clean (TCL1)
  - Use Case Make:Make Executables (TCL1)
- Feature Make:Call Tools (TCL1)
- Feature Make:Dependencies (TCL1)

Total: 6

Supported Input / Outputs

For the selected features specify the supported artifacts

- Artifact Coverage Report:SVNFile
- Artifact Executable
- Artifact Library:SVNFile
- Artifact Logfile:SVNFile
- Artifact Makefile:SVNFile
- Artifact Mapfile
- Artifact Object Code

Errors

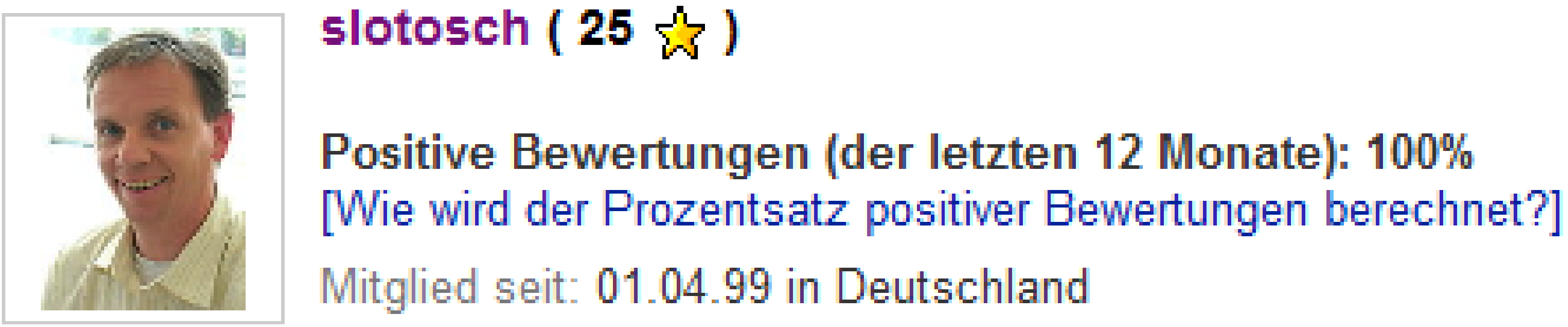
For the selected features specify the potential error classes.
The existing errors can be found at www.....

- Error Make.Make Executables:Make Builds Wrong Binary (HIGH)
- Error Make.Make Executables:Make Modifies Data (HIGH)
- Error Make.Make Executables:Old Binary Unchanged (HIGH)
- *Inferred Feature Error Make Used Wrongly in Call Tools in Make Executables (HIGH)*
- *Inferred Feature Error Make Used Wrongly in Dependencies in Make Executables (HIGH)*
- *Inferred Feature Error Make Used Wrongly in Dependencies in Make PIL in Make Executables (HIGH)*
- *Inferred Feature Error Make Used Wrongly in Dependencies in Make SIL in Make Executables (HIGH)*

Proposed Role: Eclipse Validator

There is much (different) work to do such that we need a new kind of worker: The Validator

- **Should provide confidence**
- **Should be more formalized than a committer**
- **Should have qualifications e.g. by filling out questionnaires on**
  - Eclipse qualification process
  - DO-330
- **Should have responsibilities (answer to questions)**
- **Should earn “credits” for each successful validation action**
  - Executed reviews
  - Formulated requirements
  - Created use/test cases
  - Feedback
  - …
- **Comparable:**
  Confidence in ebay:



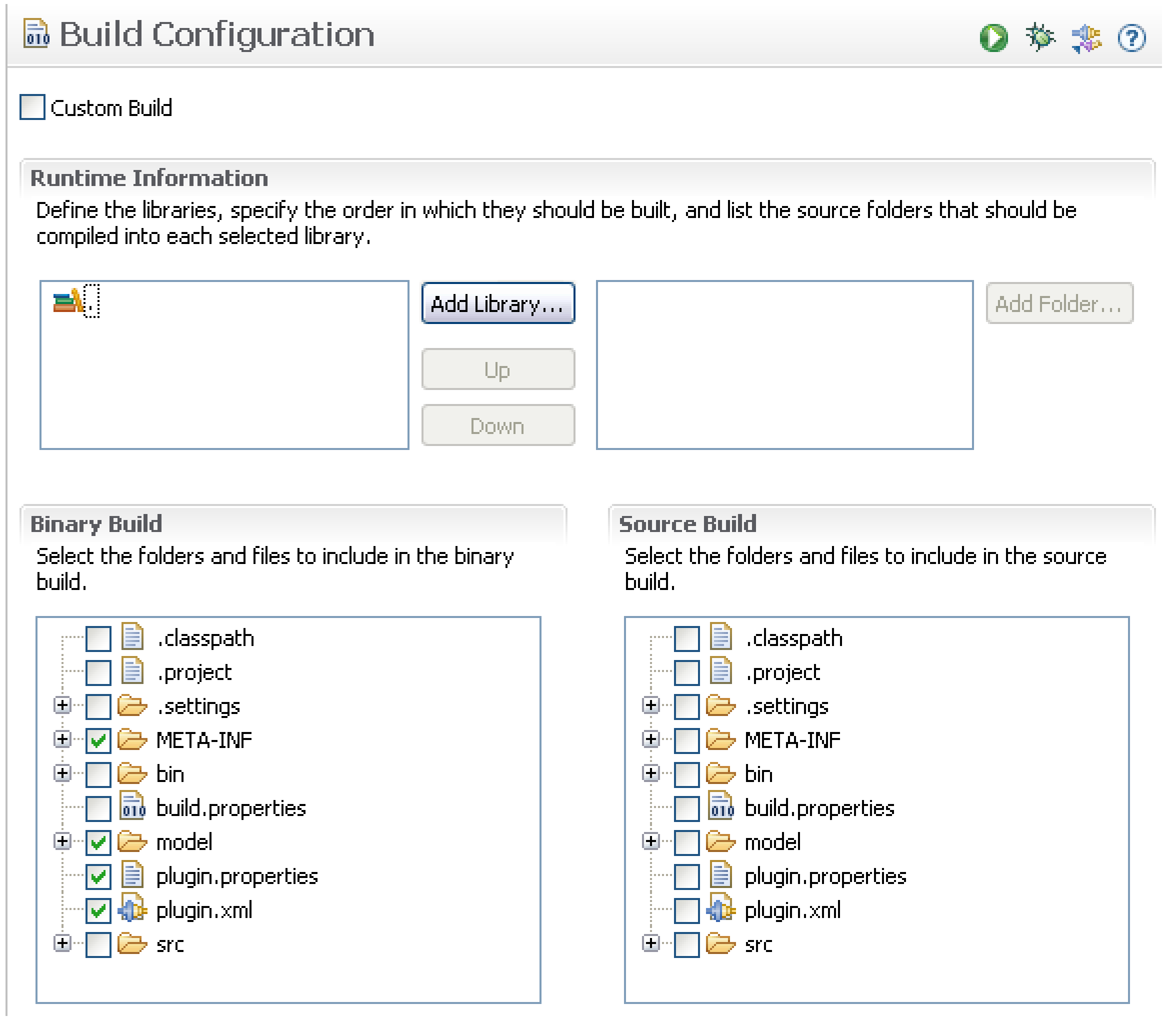
slotosch (25 ☆)

Positive Bewertungen (der letzten 12 Monate): 100%
[Wie wird der Prozentsatz positiver Bewertungen berechnet?]

Mitglied seit: 01.04.99 in Deutschland

3rd Build: Qualification Kit

- **Currently: 2 Builds available in Eclipse**
  - Source Build
  - Binary Build
- **Missing: Qualifiable Build Configuration with plugin specific**
  - Qualification information (TQL, DO-330 Model)
  - Test Cases / Coverage
  - Verification results
  - Documents
  - Involved Validators
  - …



Build Configuration

□ Custom Build

Runtime Information

Define the libraries, specify the order in which they should be built, and list the source folders that should be compiled into each selected library.

Binary Build

Select the folders and files to include in the binary build.

- □ .classpath
- □ .project
- □ .settings
- ☑ META-INF
- □ bin
- □ build.properties
- ☑ model
- ☑ plugin.properties
- ☑ plugin.xml
- □ src

Source Build

Select the folders and files to include in the source build.

- □ .classpath
- □ .project
- □ .settings
- □ META-INF
- □ bin
- □ build.properties
- □ model
- □ plugin.properties
- □ plugin.xml
- □ src

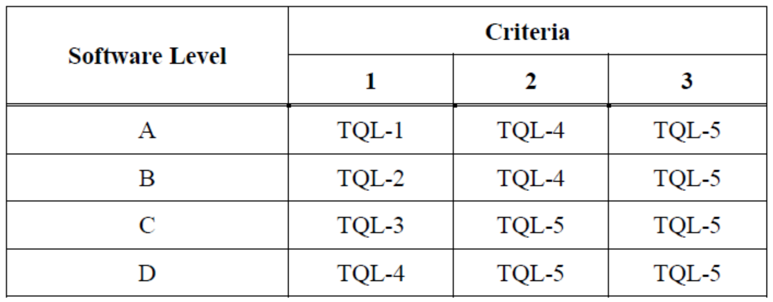
Content

- **Tool Qualification Requirements from Standards**
- **Tool Qualification Roadmap**
  - Vision
  - DO-330
  - Concept
    - Model-based Tool Qualification
    - Examples
      - Processes
      - Documents
    - Status: May 2012
- **Current Demonstration Examples**
- **Summary**

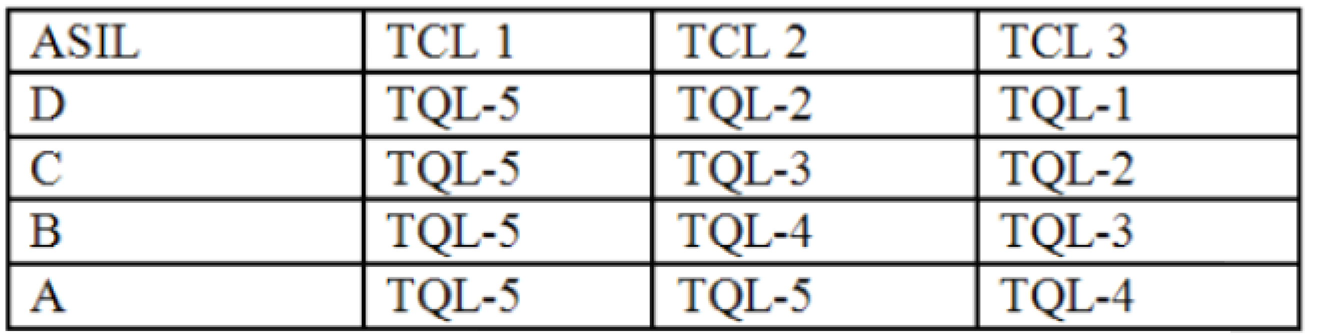
DO-330: Software Tool Qualification Considerations

- Is a safety standard applicable to all domains
- Has Tool Qualification Levels (TQL)s: TQL-1 (High), TQL-5 (Low)
- TQL-Level has to be defined from domain standards

Table 12-1 Tool Qualification Level Determination



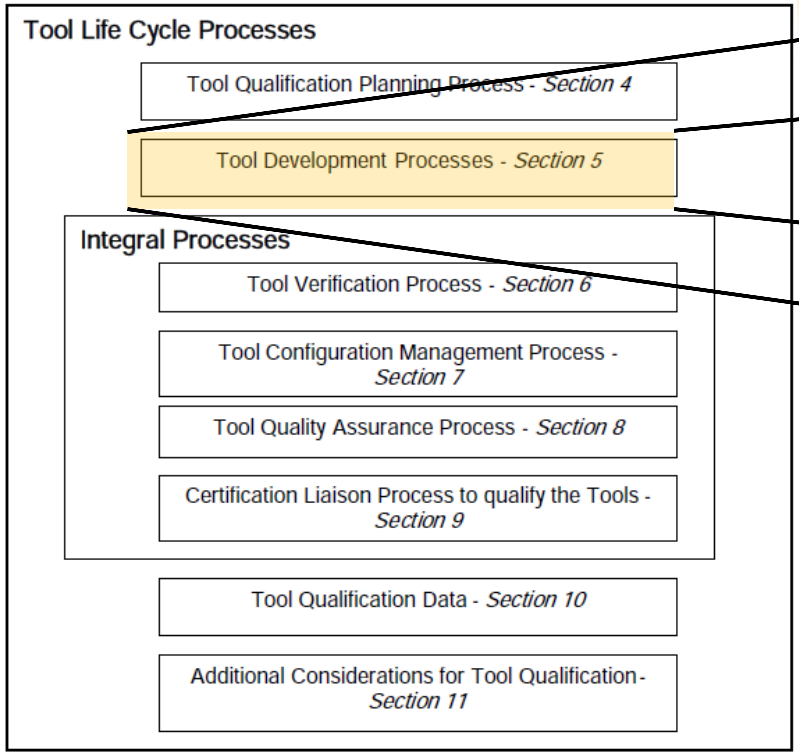
Software Level	Criteria		
	1	2	3
A	TQL-1	TQL-4	TQL-5
B	TQL-2	TQL-4	TQL-5
C	TQL-3	TQL-5	TQL-5
D	TQL-4	TQL-5	TQL-5



ASIL	TCL 1	TCL 2	TCL 3
D	TQL-5	TQL-2	TQL-1
C	TQL-5	TQL-3	TQL-2
B	TQL-5	TQL-4	TQL-3
A	TQL-5	TQL-5	TQL-4

Table 3: Determination of Tool Qualification Levels for DO-330

- **Requires**
  - Processes,
  - Activities and
  - Documents


Tool Life Cycle Processes
Tool Qualification Planning Process - Section 4
Tool Development Processes - Section 5
Integral Processes
Tool Verification Process - Section 6
Tool Configuration Management Process - Section 7
Tool Quality Assurance Process - Section 8
Certification Liaison Process to qualify the Tools - Section 9
Tool Qualification Data - Section 10
Additional Considerations for Tool Qualification - Section 11

Tool Development Processes (5.2):
- Tool Requirement Process
- Tool Design Process
- Tool Coding Process
- Tool Integration Process

10.1.1: T f
Certifica
10.1.2: T
10.1.3: T
10.1.4: T
10.1.5: T
10.1.6: T
10.1.7: T
10.1.8: Tool Design Standards
10.1.9: Tool Code Standards
10.1.10: Tool Life Cycle Environment Configuration Index
10.1.11: Tool Configuration Index
10.1.12: Tool Problem Reports
10.1.13: Tool Configuration Management Records
10.1.14: Tool Quality Assurance Records
10.1.15: Tool Accomplishment Summary
10.1.16: Tool-Specific Information in Software Accomplishment Summary
10.1.17: Tool-Specific Information in Software Life Cycle Environment Configuration Index (SECI)
…

Content

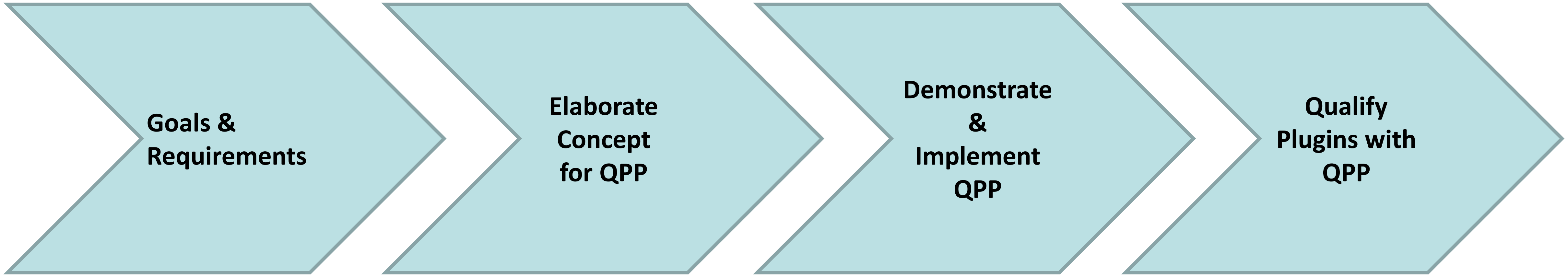
- **Tool Qualification Requirements from Standards**
- **Tool Qualification Roadmap**
  - Vision
  - DO-330
  - Concept
    - Model-based Tool Qualification
    - Roadmap Processes
    - Examples
      - Processes
      - Documents
    - Status: May 2012
- **Current Demonstration Examples**
- **Summary**

Concept for Eclipse Project QPP

- **Prepares Eclipse Project for Qualifiable Plugin Projects (QPP)**
- **Uses a separate EMF-Model (DO-330-model) for prototyping**
- **Covers the complete DO-330 (bi-directional tracing)**
  - How-To-Qualify-Document (with DO-IDs)
  - Generic Documents
    - Tool Development Plan
    - Tool Verification Plan
    - …
- **Is developed within WP5: Tool Qualification in Automotive Industrial Working Group, see http://wiki.eclipse.org/Auto_IWG_WP5**
- **Roadmap:**
  - Goal: DO-330
  - Every two weeks: new steps (process for DO-330)
  - Presented and discussed in Telcos

Roadmap to the Concept/Project QPP

1. Identify goals & requirements for tool qualification in Eclipse
2. Propose process / project (Concept)
3. Demonstrate & implement proposal
4. Establish proposal: Qualify (selected) plugins



Goals & Requirements

Elaborate Concept for QPP

Demonstrate & Implement QPP

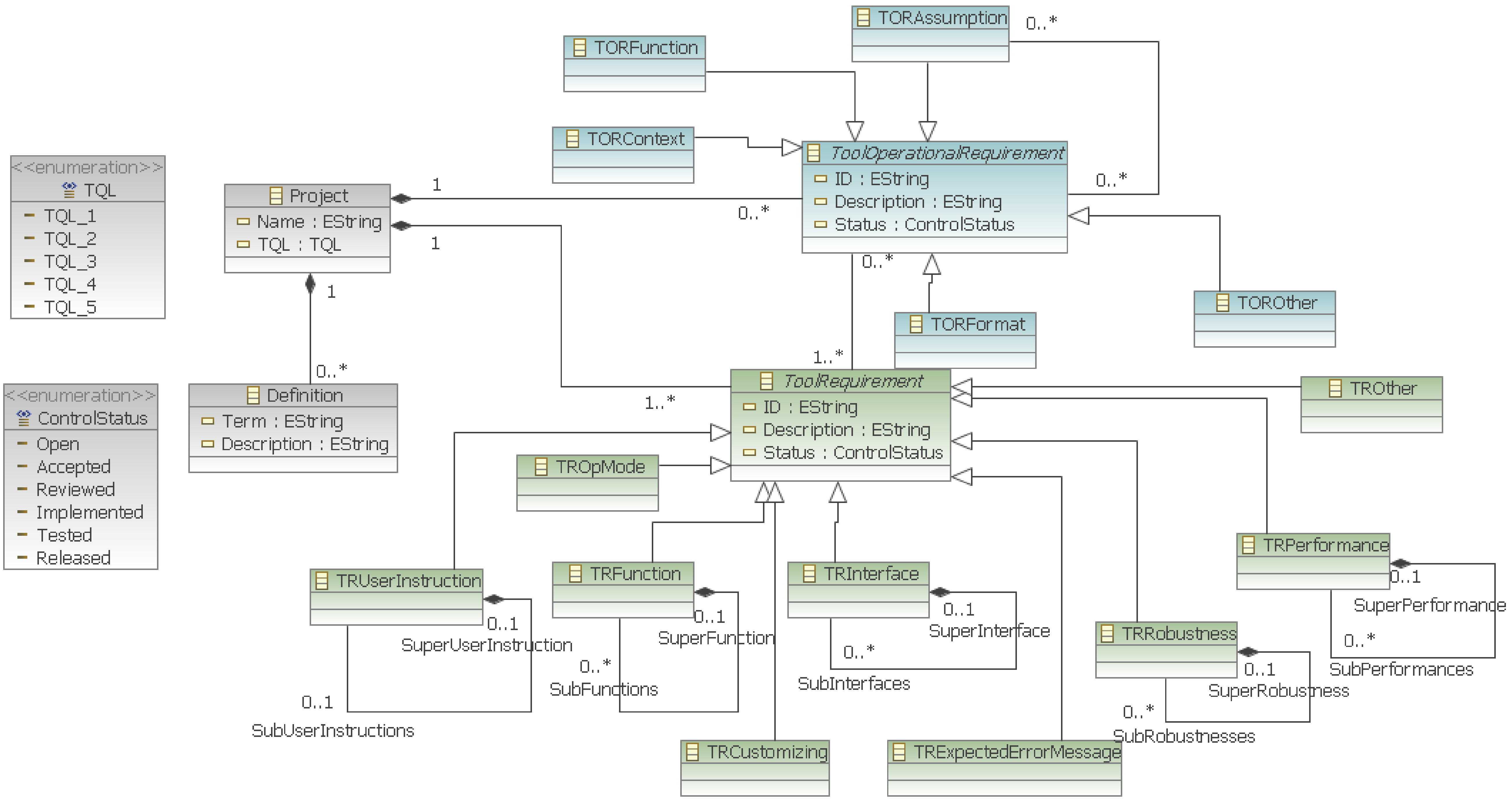
Qualify Plugins with QPP

Content

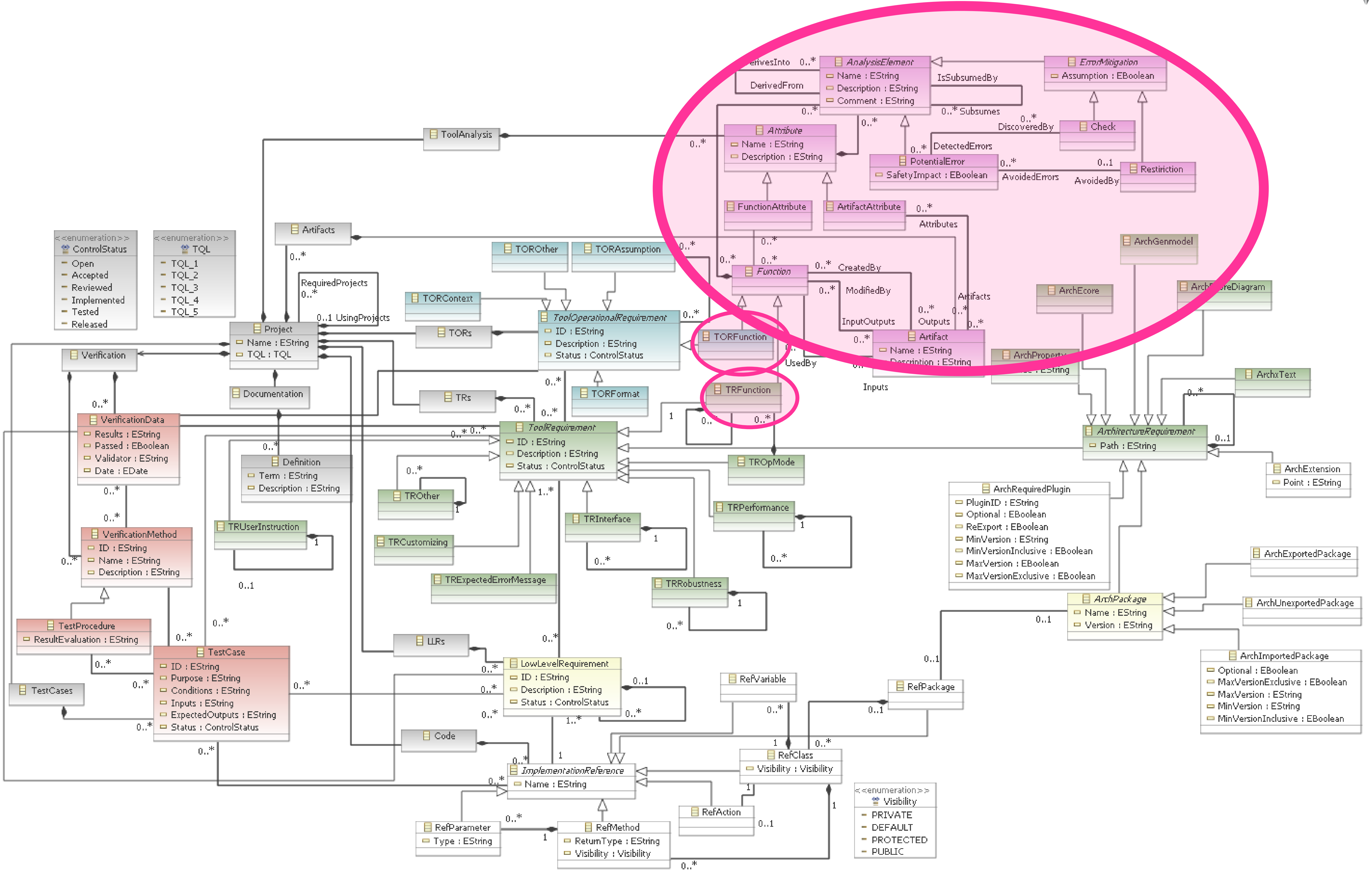
- **Tool Qualification Requirements from Standards**
- **Tool Qualification Roadmap**
  - Vision
  - DO-330
  - Eclipse Project: Qualifiable Plugin Projects (QPP)
  - Concept
    - Model Based Qualification
    - Examples
      - Processes
      - Documents
    - Status: May 2012
- **Current Demonstration Examples**
- **Summary**

Model for Tool-Requirements

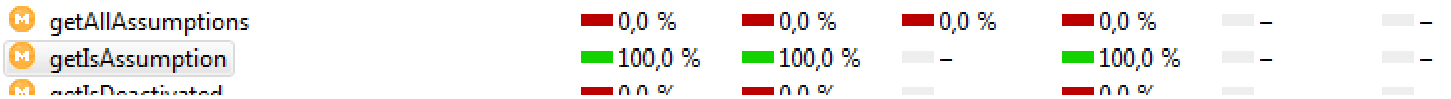
- **EMF-Metamodel for Tool Requirements**



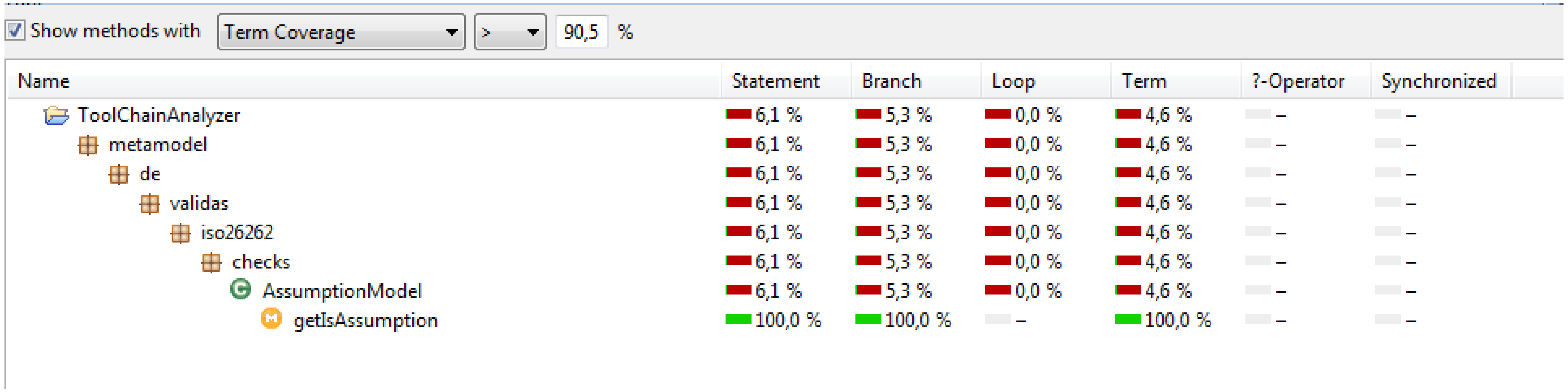
Planning: Analysis Model for PSAC



Test Coverage and Measurement

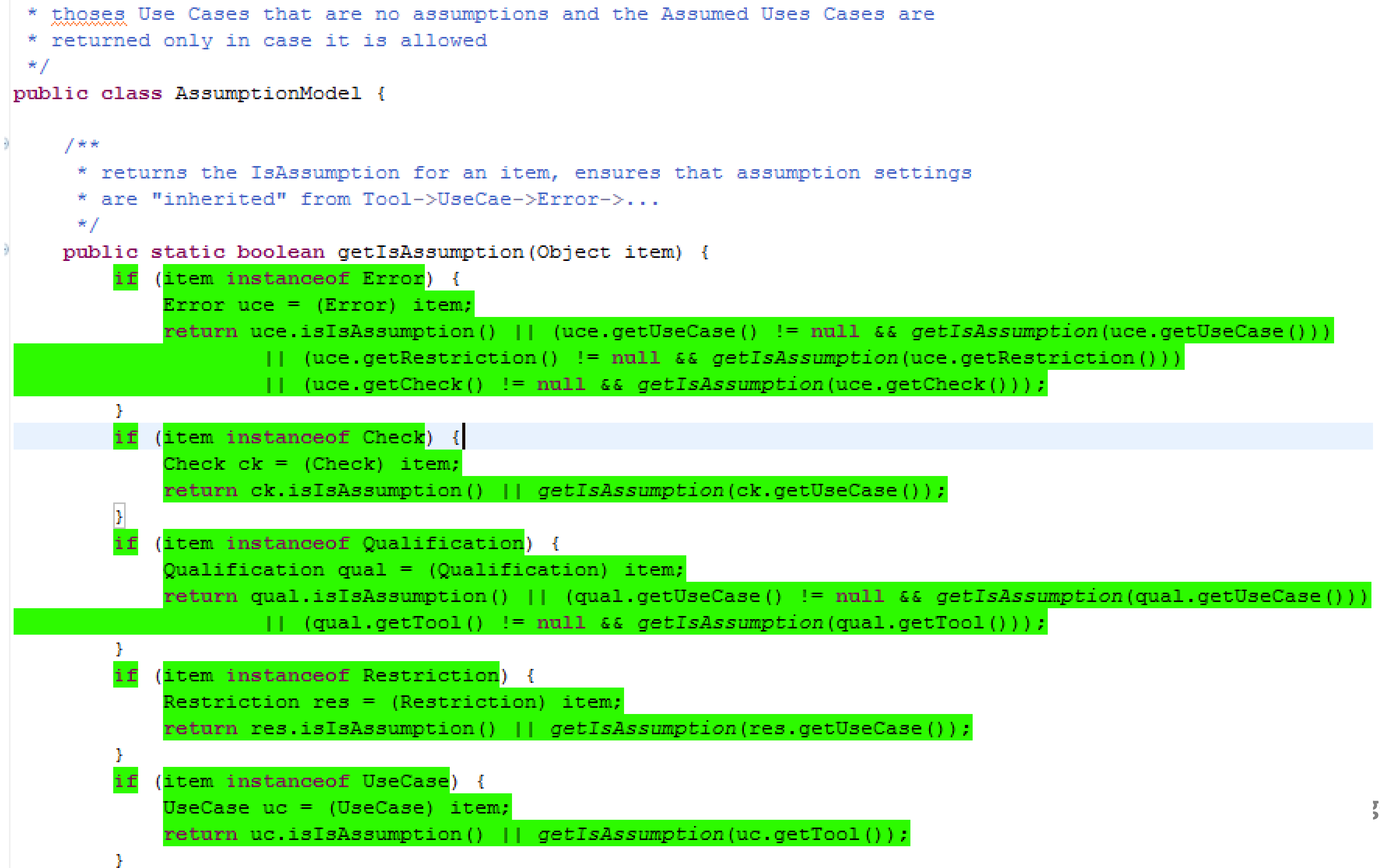


getAllAssumptions	🟥 0,0 %	🟥 0,0 %	🟥 0,0 %	🟥 0,0 %	–	–
getIsAssumption	🟩 100,0 %	🟩 100,0 %	–	🟩 100,0 %	–	–
getIsDeactivated	🟥 0,0 %	🟥 0,0 %		🟥 0,0 %		



☑ Show methods with Term Coverage ▾ > ▾ 90,5 %

Name	Statement	Branch	Loop	Term	?-Operator	Synchronized
ToolChainAnalyzer	🟥 6,1 %	🟥 5,3 %	🟥 0,0 %	🟥 4,6 %	–	–
metamodel	🟥 6,1 %	🟥 5,3 %	🟥 0,0 %	🟥 4,6 %	–	–
de	🟥 6,1 %	🟥 5,3 %	🟥 0,0 %	🟥 4,6 %	–	–
validas	🟥 6,1 %	🟥 5,3 %	🟥 0,0 %	🟥 4,6 %	–	–
iso26262	🟥 6,1 %	🟥 5,3 %	🟥 0,0 %	🟥 4,6 %	–	–
checks	🟥 6,1 %	🟥 5,3 %	🟥 0,0 %	🟥 4,6 %	–	–
AssumptionModel	🟥 6,1 %	🟥 5,3 %	🟥 0,0 %	🟥 4,6 %	–	–
getIsAssumption	🟩 100,0 %	🟩 100,0 %	–	🟩 100,0 %	–	–



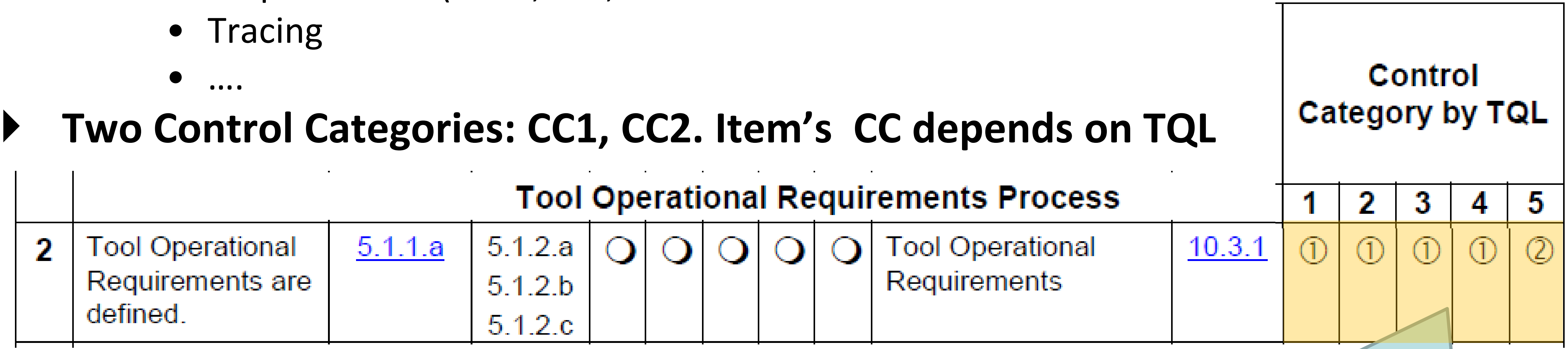
```
 * thoses Use Cases that are no assumptions and the Assumed Uses Cases are
 * returned only in case it is allowed
 */
public class AssumptionModel {

    /**
     * returns the IsAssumption for an item, ensures that assumption settings
     * are "inherited" from Tool->UseCae->Error->...
     */
    public static boolean getIsAssumption(Object item) {
        if (item instanceof Error) {
            Error uce = (Error) item;
            return uce.isIsAssumption() || (uce.getUseCase() != null && getIsAssumption(uce.getUseCase()))
                    || (uce.getRestriction() != null && getIsAssumption(uce.getRestriction()))
                    || (uce.getCheck() != null && getIsAssumption(uce.getCheck()));
        }
        if (item instanceof Check) {
            Check ck = (Check) item;
            return ck.isIsAssumption() || getIsAssumption(ck.getUseCase());
        }
        if (item instanceof Qualification) {
            Qualification qual = (Qualification) item;
            return qual.isIsAssumption() || (qual.getUseCase() != null && getIsAssumption(qual.getUseCase()))
                    || (qual.getTool() != null && getIsAssumption(qual.getTool()));
        }
        if (item instanceof Restriction) {
            Restriction res = (Restriction) item;
            return res.isIsAssumption() || getIsAssumption(res.getUseCase());
        }
        if (item instanceof UseCase) {
            UseCase uc = (UseCase) item;
            return uc.isIsAssumption() || getIsAssumption(uc.getTool());
        }
```

Tool Life Cycle for Qualifiable Plugins

- **Combines the following processes:**
  - Planning (TORs)
  - Development (TR, LLRs)
  - Integration (Verification)
  - Configuration Management
  - Quality Assurance
- **Fits to existing processes (Project process, Release Process) by extending them with a “Qualification Stage”**
- **The following stages are defined (and can be determined automatically from the DO-330 model) such that every release has a well-defined qualification stage**
  - **Unqualified-Pre-Alpha Release** (“**Undefined**”): unknown qualification state
  - **Qualification Alpha-Release** (“**Analyzed**”): The TORs are defined and TQL is determined
  - **Qualification Beta-Release** (“**Feature-Complete**”): All requirements (TORs and TRs) are described and have traces to LLRs and Code
  - **Qualification Release Candidate** (“**Verification Defined**”): All required verification steps are defined. No open bugs of the category “Blocker” are available.
  - **Qualification Release**: (“**Successfully Verified**”) Verification has been successfully executed and are documented within the qualification kit
- **Transition Criteria are formally defined, based on the DO-330 model**

Configuration Management

- Configuration Items are all elements within the Qualifiable Eclipse Project
  - Sources
  - Architecture
  - DO-330-model
    - Requirements (TORs, TRs,
    - Tracing
    - ….
- Two Control Categories: CC1, CC2. Item's CC depends on TQL



											Control Category by TQL				
	Tool Operational Requirements Process										1	2	3	4	5
2	Tool Operational Requirements are defined.	5.1.1.a	5.1.2.a 5.1.2.b 5.1.2.c	❍	❍	❍	❍	❍	Tool Operational Requirements	10.3.1	①	①	①	①	②

- Definition of Control Categories (DO-330):

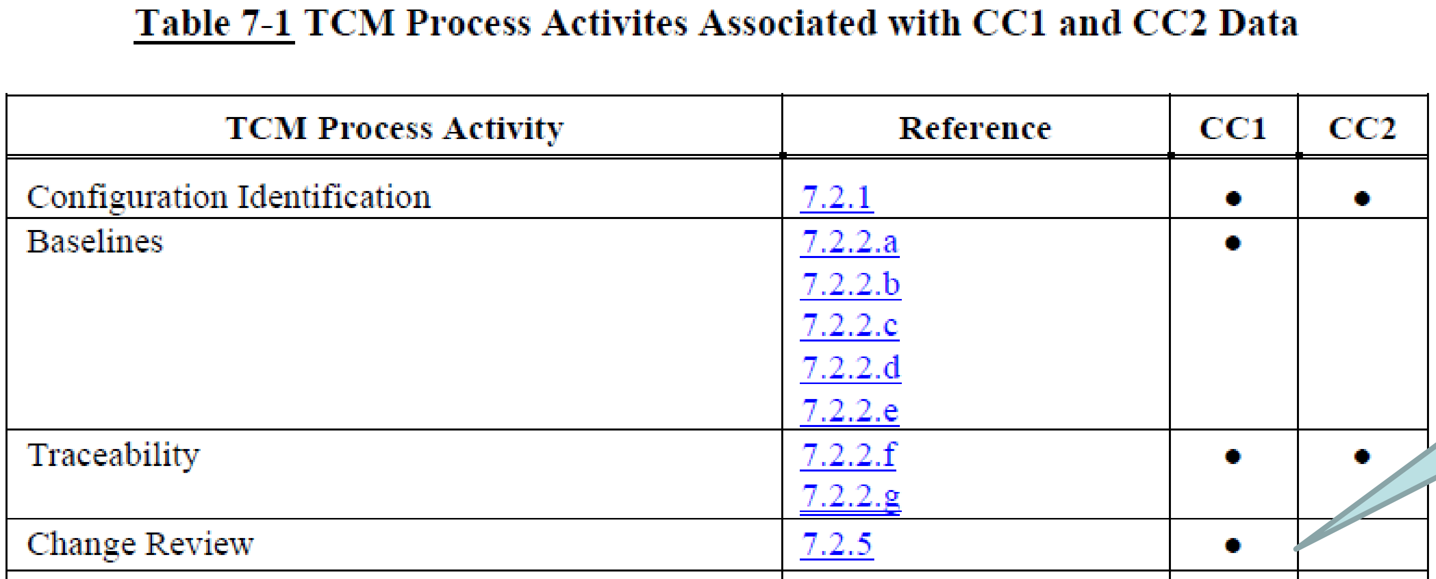


Table 7-1 TCM Process Activites Associated with CC1 and CC2 Data

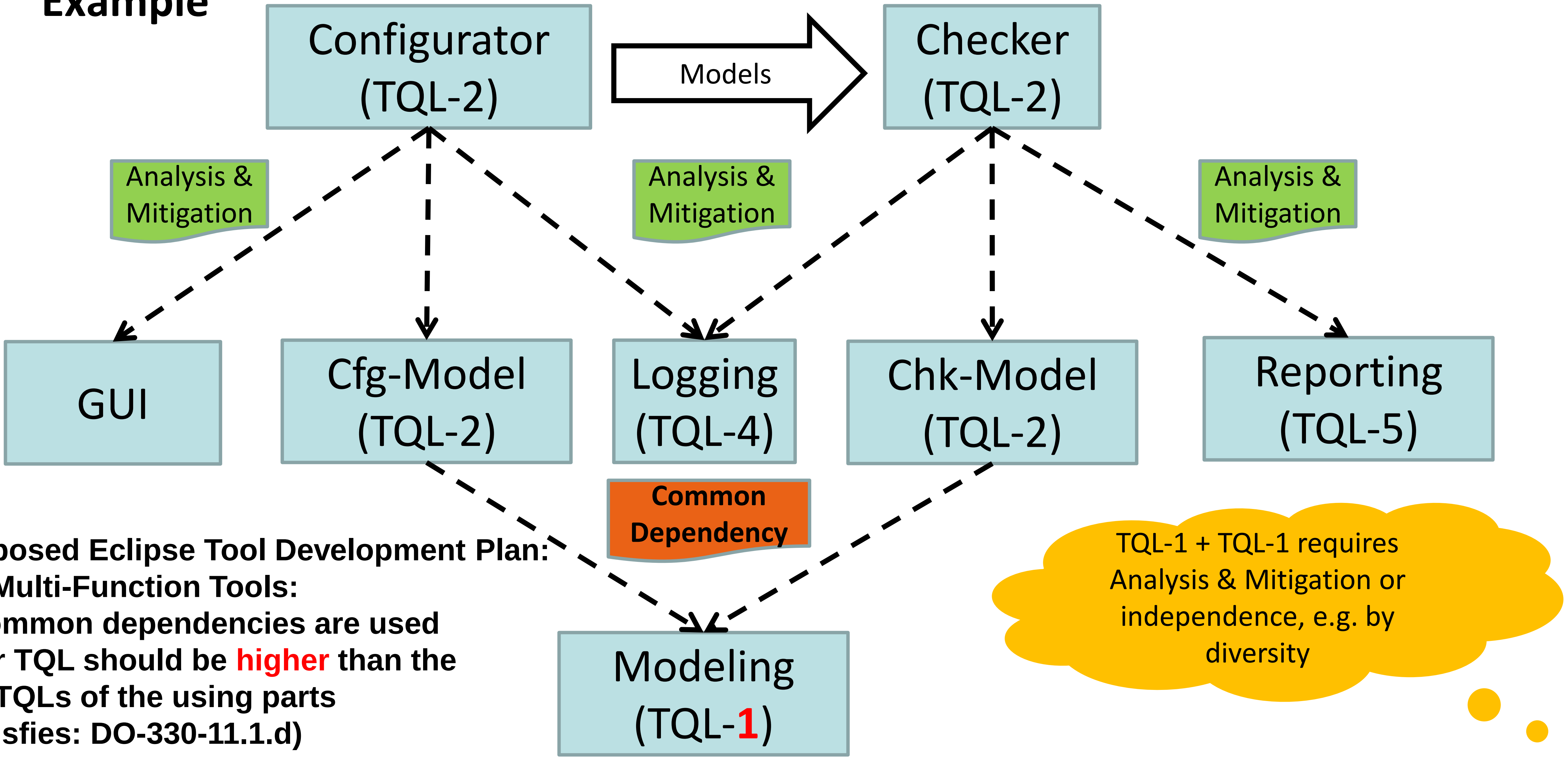
TCM Process Activity	Reference	CC1	CC2
Configuration Identification	7.2.1	•	•
Baselines	7.2.2.a 7.2.2.b 7.2.2.c 7.2.2.d 7.2.2.e	•	
Traceability	7.2.2.f 7.2.2.g	•	•
Change Review	7.2.5	•	

Example: TORs **changes** have to be **reviewed** for TQL-1 to TQL-4 but not for TQL-5

Plugin Extension has to know this (Transition Criteria!)

Independence Challenge in MF Tools

- **11.1.d (Additional Considerations for Multi-Function Tools) states: if multi-function tools both produce and verify the same output**
  - Protection shall be used (plugins) AND
  - (for TQL-1 and TQL-2) **Independence**
- **Example**

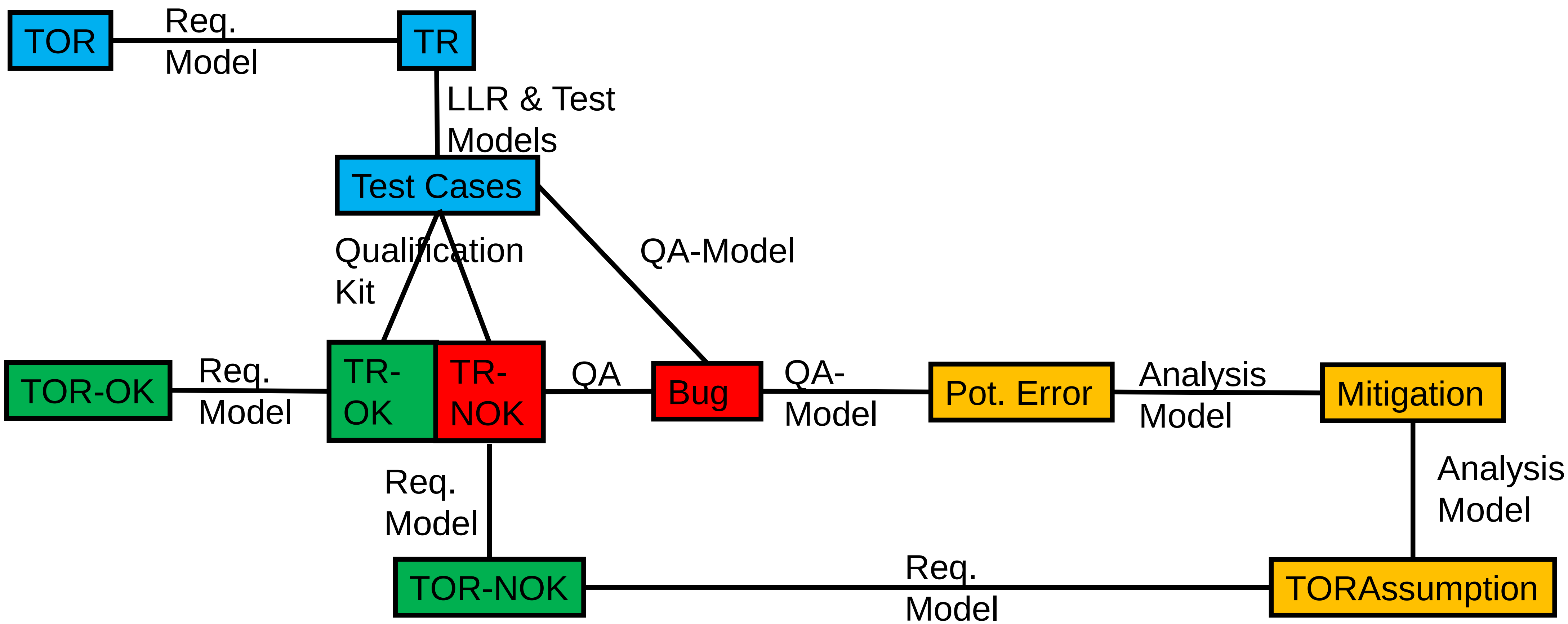


TQL-1 + TQL-1 requires Analysis & Mitigation or independence, e.g. by diversity

Proposed Eclipse Tool Development Plan:
3.2 Multi-Function Tools:
If common dependencies are used their TQL should be higher than the TQLs of the using parts
(satisfies: DO-330-11.1.d)

Qualification Liaison Process

- **For all tools with qualification need**
- **Demonstrate that the tools conform to their requirements (“TOR”), even if qualification shows errors**



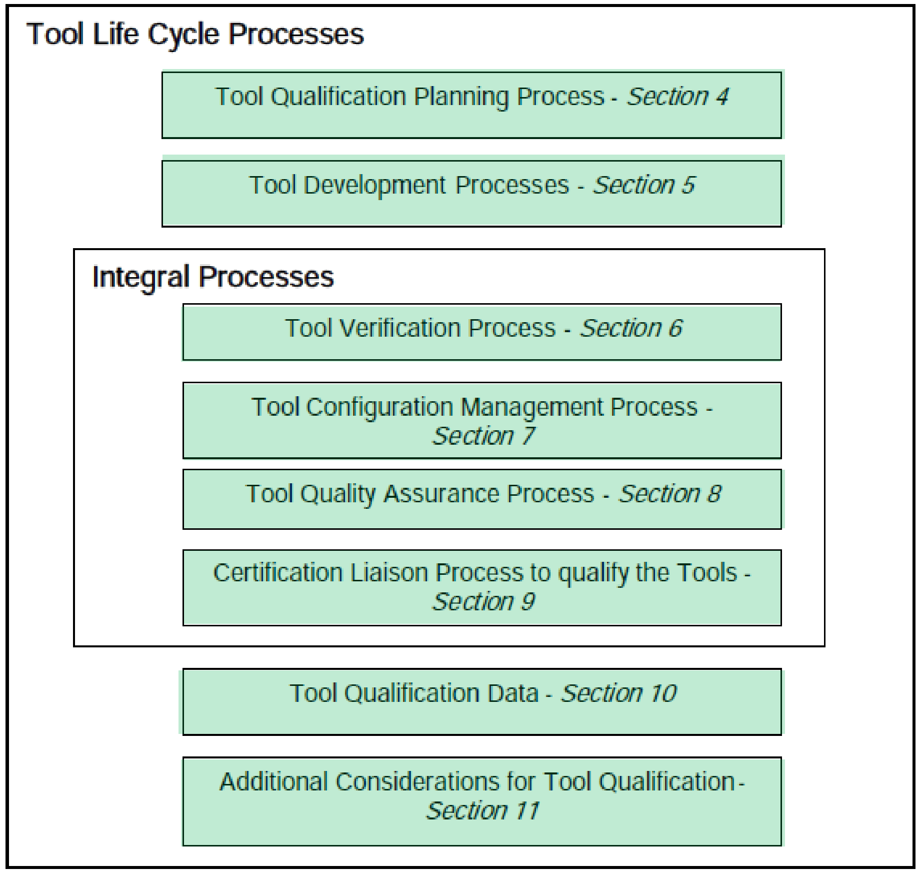
Content

- **Tool Qualification Requirements from Standards**
- **Tool Qualification Roadmap**
  - Vision
  - DO-330
  - Concept
    - Model Based Qualification
    - Examples
      - Processes
      - Documents
    - Status: May 2012
- **Current Demonstration Examples**
- **Summary**

Roadmap - Status May 2012

1. Goals: DO-330
2. Concept: Eclipse Project QPP
3. Demonstrate & implement QPP
4. Qualify (selected) plugins

- **Status May 2012**



Tool Life Cycle Processes

Tool Qualification Planning Process - *Section 4*

Tool Development Processes - *Section 5*

Integral Processes

Tool Verification Process - *Section 6*

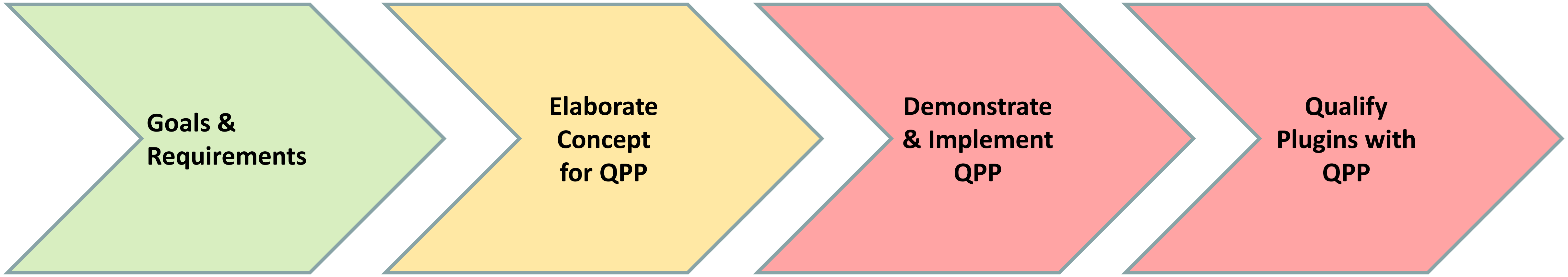
Tool Configuration Management Process - *Section 7*

Tool Quality Assurance Process - *Section 8*

Certification Liaison Process to qualify the Tools - *Section 9*

Tool Qualification Data - *Section 10*

Additional Considerations for Tool Qualification - *Section 11*



Goals & Requirements

Elaborate Concept for QPP

Demonstrate & Implement QPP

Qualify Plugins with QPP

Identified

Status: Feasible and 100% of DO-330 covered

Ready to start

Ready to start (Prototyping)

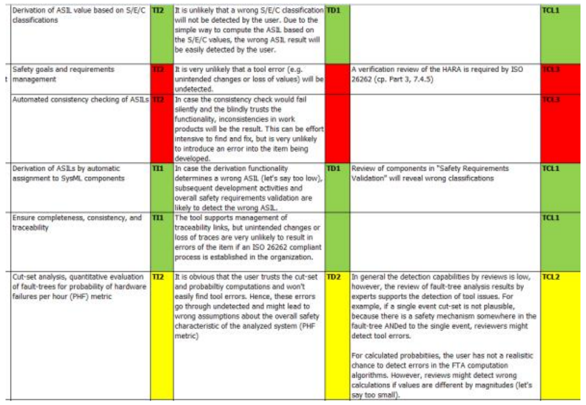
- **Summary: Qualification is feasible and qualification (based on current prototype) could be started now**

Content

- **Tool Qualification Requirements from Standards**
- **Tool Qualification Roadmap**
  - Vision
  - DO-330
  - Concept
    - Model Based Qualification
    - Examples
      - Processes
      - Documents
    - Status: May 2012
- **Current Demonstration Examples**
- **Summary**

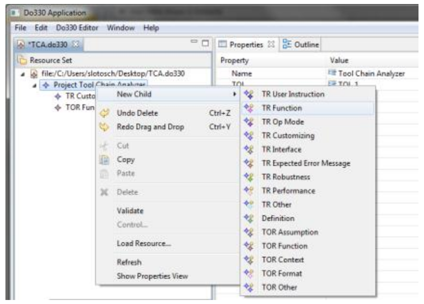
Demonstration / Examples

- medini analyze **Tool/Process Analysis & TCL Determination in Progress (work in progress)**



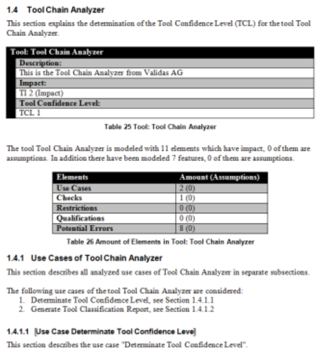
Derivation of ASIL value based on S/E/C classifications	TI2	It is unlikely that a wrong S/E/C classification will not be detected by the user. Due to the simple way to compute the ASIL based on the S/E/C values, the wrong ASIL result will be easily detected by the user.	TD1		TCL1
Safety goals and requirements management	TI2	It is very unlikely that a tool error (e.g. unintended changes or loss of values) will be undetected.		A verification review of the HARA is required by ISO 26262 (cp. Part 3, 7.4.5)	TCL3
Automated consistency checking of ASILs	TI2	In case the consistency check would fail silently and the blindly trusts the functionality, inconsistencies in work products will be the result. This can be effort intensive to find and fix, but is very unlikely to introduce an error into the item being developed.			TCL3
Derivation of ASILs by automatic assignment to SysML components	TI1	In case the derivation functionality determines a wrong ASIL (let's say too low), subsequent development activities and overall safety requirements validation are likely to detect the wrong ASIL.	TD1	Review of components in "Safety Requirements Validation" will reveal wrong classifications	TCL1
Ensure completeness, consistency, and traceability	TI1	The tool supports management of traceability links, but unintended changes or loss of traces are very unlikely to result in errors of the item if an ISO 26262 compliant process is established in the organization.			TCL1
Cut-set analysis, quantitative evaluation of fault-trees for probability of hardware failures per hour (PHF) metric	TI2	It is obvious that the user trusts the cut-set and probabiltiy computations and won't easily find tool errors. Hence, these errors go through undetected and might lead to wrong assumptions about the overall safety characteristic of the analyzed system (PHF metric)	TD2	In general the detection capabilities by reviews is low, however, the review of fault-tree analysis results by experts supports the detection of tool issues. For example, if a single event cut-set is not plausible, because there is a safety mechanism somewhere in the fault-tree ANDed to the single event, reviewers might detect tool errors. For calculated probabitiies, the user has not a realisitic chance to detect errors in the FTA computation algorithms. However, reviews might detect wrong calculations if values are different by magnitudes (let's say too small).	TCL2

- VALIDAS **Tool Chain Analyzer: DO-330 Modeling of Analysis, TCL, Requirements, Design, Tests including document templates (for generation)**



Property	Value
Name	Tool Chain Analyzer
TQL	TQL 1

Generate Documents



1.4 Tool Chain Analyzer

This section explains the determination of the Tool Confidence Level (TCL) for the tool Tool Chain Analyzer.

Tool: Tool Chain Analyzer
Description:
This is the Tool Chain Analyzer from Validas AG
Impact:
TI 2 (Impact)
Tool Confidence Level:
TCL 1

Table 25 Tool: Tool Chain Analyzer

The tool Tool Chain Analyzer is modeled with 11 elements which have impact, 0 of them are assumptions. In addition there have been modeled 7 features, 0 of them are assumptions.

Elements	Amount (Assumptions)
Use Cases	2 (0)
Checks	1 (0)
Restrictions	0 (0)
Qualifications	0 (0)
Potential Errors	8 (0)

Table 26 Amount of Elements in Tool: Tool Chain Analyzer

1.4.1 Use Cases of Tool Chain Analyzer

This section describes all analyzed use cases of Tool Chain Analyzer in separate subsections.

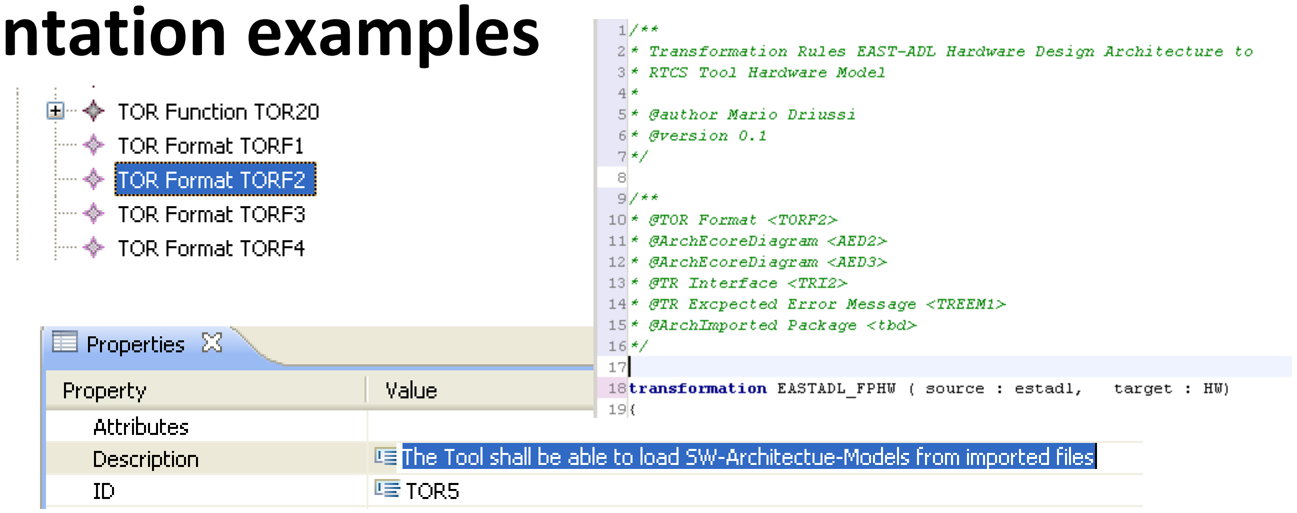
The following use cases of the tool Tool Chain Analyzer are considered:

1. Determinate Tool Confidence Level, see Section 1.4.1.1
2. Generate Tool Classification Report, see Section 1.4.1.2

1.4.1.1 [Use Case Determinate Tool Confidence Level]

This section describes the use case "Determinate Tool Confidence Level".

- virtual vehicle **Real-time system configuration tool for AUTOSAR: DO-330 model and implementation examples**



```
/**
* Transformation Rules EAST-ADL Hardware Design Architecture to
* RTCS Tool Hardware Model
*
* @author Mario Driussi
* @version 0.1
*/

/**
* @TOR Format <TORF2>
* @ArchEcoreDiagram <AED2>
* @ArchEcoreDiagram <AED3>
* @TR Interface <TRI2>
* @TR Excpected Error Message <TREEM1>
* @ArchImported Package <tbd>
*/

transformation EASTADL_FPHW ( source : estadl,   target : HW)
{
```

Property	Value
Attributes	
Description	The Tool shall be able to load SW-Architectue-Models from imported files
ID	TOR5

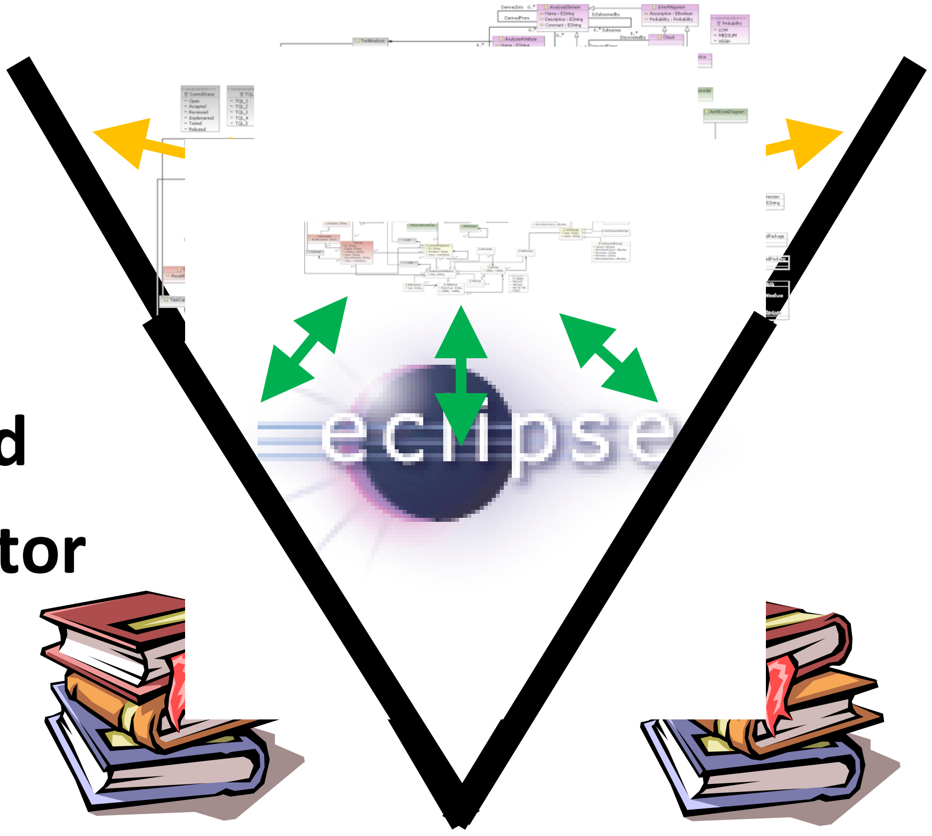
- …

Content

- **Tool Qualification Requirements from Standards**
- **Tool Qualification Roadmap**
  - Vision
  - DO-330
  - Concept
    - Model Based Qualification
    - Examples
      - Processes
      - Documents
    - Status: May 2012
- **Current Demonstration Examples**
- **Summary**

Summary

- **Extended Eclipse (QPP) will support qualification including**
  - Classification: Tool Analysis -> Planning Process
  - Qualification: Process & Model for qualifiable plugin projects
  - Usage: Fulfill assumptions and apply qualification kits
- **Applicable to all relevant standards (ISO 26262, IEC 61508, DO-178C, EN 50128,..)**
- **Metadata extension for qualification information of plugins: DO-330 model**
- **Much work in progress**
  - Tracing to "How-To-Qualify" document
  - Modeling: gaps to current meta-information
  - Create documentations (TDP,TVP,TQP,TQR..)
- **First, second, third, fourth, fifth steps performed**
- **Proposed new role for that work: Eclipse Validator**
- **Many areas of DO-330 already covered**



Thank You!

VALIDAS

Arnulfstraße 27
80335 München
www.validas.de
info@validas.de