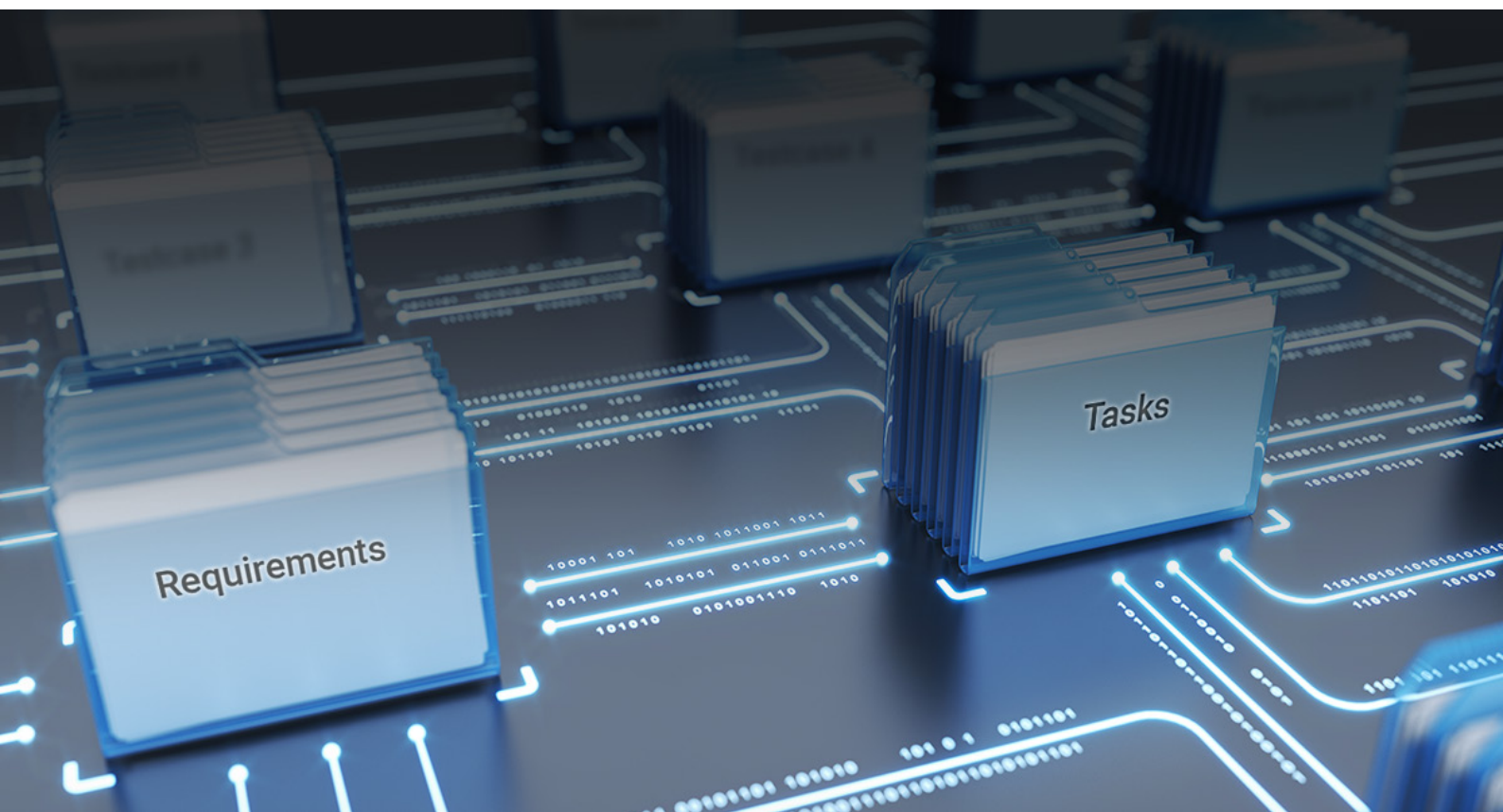


codebeamer & TPT

Whitepaper



Whitepaper codebeamer ALM & TPT

Abstract

This white paper gives insights into the integration of codebeamer and the testing tool TPT. Against the backdrop of this integration the importance of the traceability of requirements and test cases is illustrated.

In particular, an introduction to codebeamer and the testing tool TPT is given. Moreover, the importance of requirements-based testing is stressed and the development process throughout the release cycles with codebeamer and TPT is described.

Introduction

Throughout the development process the engineer is facing a myriad of tasks, two of which play a central role. Requirements need to be implemented and test cases need to be created and executed.

Moreover, running test cases in order to verify that implementations match the requirements is also done for quality assurance purposes.

Lastly, the implementation of all requirements needs to be checked against the corresponding hardware integration and vehicle integration level.

Since safety norms (e.g. ISO 26262) demand traceability of requirements, an integration of all of the above steps is not only of organizational but also of legal importance.

This white paper illustrates the test process using codebeamer and the testing tool TPT

Introduction codebeamer

codebeamer is an Engineering and Application Lifecycle Management (EALM) platform for advanced product and software development. It was designed to help create better products in a faster and more efficient way through its unique set of features and integrations. codebeamer offers requirements management, development management, quality assurance and testing, Agile project management, DevOps, product line and risk management, powerful review features, regulatory compliance management and release management – all in an integrated, highly efficient, and user-friendly platform.

Introduction testing tool TPT

The testing tool TPT is a tool for testing embedded control software. It is used for the automatic creation, generation and execution of test cases. Moreover, all tests can be assessed and managed requirements-based.

All development phases from unit and module testing over software integration testing up to vehicle testing can be performed with TPT. All test environments are supported from MiL, SiL, PiL to HiL and automated driving tests.

The testing tool is also suitable for testing safety critical systems (ISO26262).

Relation of artefacts towards one another

Generally, test requirements describe the “shall-be behavior” of a system and the tests itself verify whether the system does what it should. Therefore, linking requirements and tests is key, and requirements-based testing plays a central role in testing with TPT. Besides, implementations are also linked to requirements but that is not the issue at stake here.

Essentially, what you want to show is that you have tested sufficiently in light of the requirements. In order to prove the completeness of your testing efforts, you’ll have to link requirements with the corresponding tests. As a result, untested requirements can be easily identified and reported.

Development through multiple release cycles

In order for the development process to run smoothly, every step of the way needs to be facilitated and supported by all means possible. This means, first of all, using an ALM tool such as codebeamer to organize releases and to manage the traceability of artefacts such as tests and requirements.

In the development process there are changes in requirements over different development cycles and releases. With TPT, changes between releases and their effect on testing can be easily tracked. This way, it is very easy to determine which requirements have changed and which test cases are affected by a new release cycle. This simplifies the process significantly.

Integrating ALM into TPT comes in quite handy when directly importing requirements and tests. You can also first develop the tests in TPT, then export them into codebeamer. In other words, the integration allows for many automated setups and proves to be very flexible in use.

Eventually, this enables a cost and time-efficient progression of development and release cycles.

The essential steps to along the automated test process are:

1. Import requirements from codebeamer into TPT
2. Import tests from codebeamer into TPT (If you already specified tests)
3. Import linking of artefacts imported in 1. and 2.
4. If applicable, create new tests in TPT or adapt existing tests and export them back to codebeamer
5. Implement tests in TPT
6. Perform tests in TPT
7. Export test results as test runs into codebeamer for further processing

In codebeamer ALM requirements are structured as shown below:

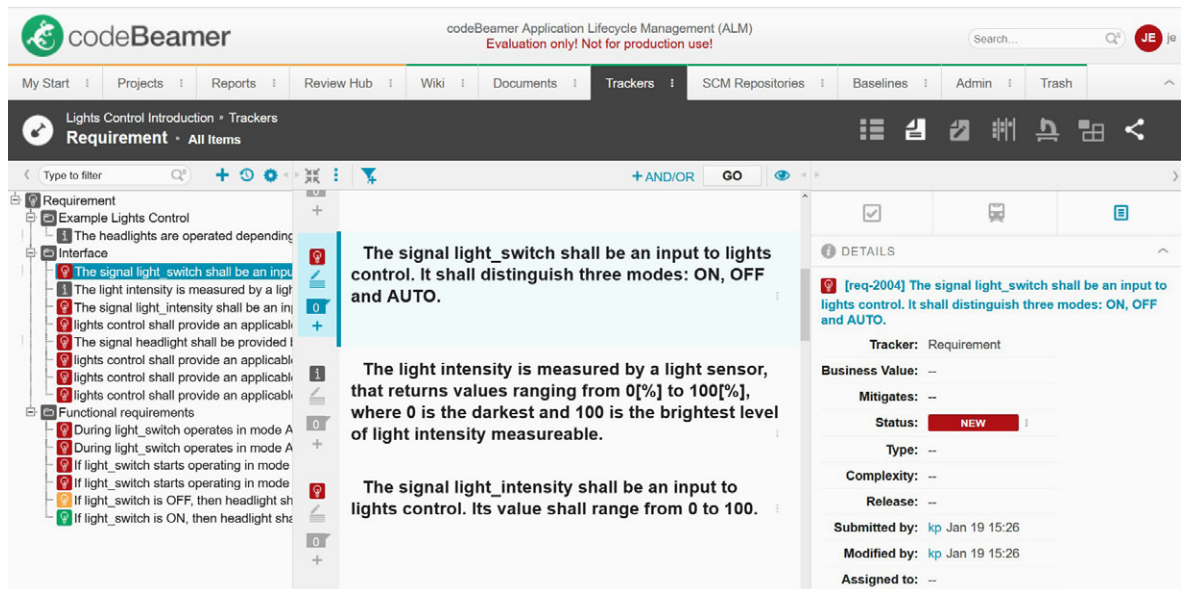


Figure 1 Requirements View in codebeamer

Requirements can be imported from requirements trackers or management platforms such as an ALM tool into TPT. Imported requirements in TPT for the given example are shown in the figure below.

Content Signature Initial Values Assesslet Content Report Requirements x Help

Requirement sets: < all requireme... 13/13 Filter ...

Link	ID	Text	# TCs	# Varia...	# Assesslets	Description	Parent	Status	Tracker	Type
<input type="checkbox"/>	200	Example Lights Control				--		New	Requirement	Folder
<input type="checkbox"/>	2002	The headlights are operated depend				--	Example Lights Cont	New	Requirement	Information
<input type="checkbox"/>	200	Interface				--		New	Requirement	Folder
<input checked="" type="checkbox"/>	2004	The signal light_switch shall be an input to lights control. It shall distinguish three modes: ON, OFF and AUTO.	4			--	Interface	New	Requirement	
<input type="checkbox"/>	2005	The light intensity is measured by a				--	Interface	New	Requirement	Information
<input type="checkbox"/>	2006	The signal light_intensity shall be an input to lights control. It shall distinguish three modes: ON, OFF and AUTO.				--	Interface	New	Requirement	
<input type="checkbox"/>	2008	lights control shall provide an application for the signal headlight shall be provided				--	Interface	New	Requirement	
<input type="checkbox"/>	2007	The signal headlight shall be provided				--	Interface	New	Requirement	
<input type="checkbox"/>	2009	lights control shall provide an application for the signal headlight shall be provided				--	Interface	New	Requirement	
<input type="checkbox"/>	2010	lights control shall provide an application for the signal headlight shall be provided				--	Interface	New	Requirement	
<input type="checkbox"/>	2011	lights control shall provide an application for the signal headlight shall be provided				--	Interface	New	Requirement	
<input type="checkbox"/>	201	Functional requirements				--		New	Requirement	Folder
<input type="checkbox"/>	2013	During light_switch operates in mode 4				--	Functional requirement	New	Requirement	
<input type="checkbox"/>	2014	During light_switch operates in mode 3				--	Functional requirement	New	Requirement	
<input checked="" type="checkbox"/>	2015	If light_switch starts operating in mode 2				--	Functional requirement	New	Requirement	
<input checked="" type="checkbox"/>	2016	If light_switch starts operating in mode 2				--	Functional requirement	New	Requirement	
<input checked="" type="checkbox"/>	2017	If light_switch is OFF, then headlight shall be ON				--	Functional requirement	Draft	Requirement	
<input checked="" type="checkbox"/>	2018	If light_switch is ON, then headlight shall be OFF				--	Functional requirement	Waiting for approval	Requirement	

Requirement details - showing requirement with ID 2004:

Attribute	Value
ID	2004
Text	The signal light_switch shall be an input to lights control. It shall distinguish three modes: ON, OFF and AUTO.
URI	http://vs7.piketec.local:3000/cb/issue/2004
Comment	
Description	--
Modified at	2021-01-19T15:26:37.879
Modified by	kp
Parent	Interface
Release	
Status	New
Submitted at	2021-01-19T15:26:37.879
Submitted by	kp
Tracker	Requirement
Type	

Linked objects Attachments of attribute 'Release' (0)

- bright ambient; AUTO 2 ON 2 OFF 2 ON
- bright ambient; OFF 2 ON 2 AUTO 2 OFF
- dark ambient; AUTO 2 OFF 2 ON 2 OFF
- dark ambient; ON 2 OFF 2 AUTO 2 ON

Figure 2 Requirements View in TPT

Once all requirements have been imported, tests can be implemented. If there are already existing test cases in the ALM tool, TPT will create and link the test cases accordingly (maintaining traceability). Next, the test implementation can then be done in TPT. New test cases that are created in TPT or existing ones that are modified can be exported to codebeamer.

codeBeamer Application Lifecycle Management (ALM)
Evaluation only! Not for production use!

My Start | Projects | Reports | Review Hub | Wiki | Documents | Trackers | SCM Repositories | Baselines | Admin | Trash

Lights Control Introduction > Trackers
Test Cases > All Items

Type to filter

Test Cases

- change switch
 - OFF ON
 - dark ambient; ON 2 OFF 2 AUTO 2 ON
 - bright ambient; OFF 2 ON 2 AUTO 2 OFF
 - dark ambient; AUTO 2 OFF 2 ON 2 OFF
 - bright ambient; AUTO 2 ON 2 OFF 2 ON
 - ON
- changing ambient
 - ON changing ambient
 - OFF changing ambient
 - AUTO changing ambient long
 - AUTO changing ambient short dark
 - AUTO changing ambient short bright
 - AUTO changing to bright full range

3 Test Steps

Critical	Action*	Expected result
#1 -	Turn Light Switch OFF	Light is OFF
#2 -	Turn Light Switch ON	Light is ON
#3 -	Turn Light Switch OFF	Light is OFF

Edit steps

dark ambient; ON 2 OFF 2 AUTO 2 ON

6 Test Steps

bright ambient; OFF 2 ON 2 AUTO 2 OFF

0 Test Steps

DETAILS

[TESTCASE-2121] OFF ON

Category: Test Cases

Priority: --

Status: NEW

Type: --

Submitted by: kp Jan 26 14:41

Modified by: je Apr 14 10:12

Assigned to: --

Verifies: If light_switch is OFF, t-hall immediately be OFF. If light_switch is ON, th~shall immediately be ON.

Pass Conditions: --

Preconditions: --

Figure 3 Test Cases in codebeamer

The screenshot displays the TPT interface with the following components:

- Project Explorer (Left):** Shows a tree view of the project structure. Under 'Lights_Control', there are 'States' and 'Test Cases [12]'. The 'Test Cases' folder is expanded, showing a list of test cases with their IDs and statuses (e.g., 'OFF ON [ID=5 status=New]', 'dark ambient; ON 2 OFF 2 AUTO 2 ON [ID=6 status=New]', etc.).
- Test Case Details (Bottom Left):** A table showing the details of the selected test case 'OFF ON'.

Attribute	Value
Test Specification	
Preconditions	
Pass Conditions	
codeBeamer URL	http://vs7.piketec.local:3000/cb/issue/2121
Submitted by	kp
Status	New
Category	Test Cases
Submitted at	2021-01-26T14:41:10.052
Parent	change switch
Test Steps	Turn Light Switch OFF, Light is OFF, false Turn Light Switch ON, Light is ON, false Turn Light Switch OFF, Light is OFF, false
Verifies	If light_switch is OFF, then headlight shall immediately be OFF., If light_switch is ON, then headlight shall immediately be ON.
Attachments	comments cannot be parsed
codeBeamer ID	2121
Name	OFF ON
Modified by	je
Modified at	2021-04-14T10:12:33.311
Reusable	false
ID	2121
- Requirements List (Right):** A table showing a list of requirements.

Link	ID	Text	# TCs	# Var
	200	Example Lights Control		
	2002	The headlights are operated depending on the light switch position and the light ir		
	200	Interface		
	2004	The signal light_switch shall be an input to lights control. It shall distinguish three r	4	
	2005	The light intensity is measured by a light sensor, that returns values ranging from 0		
	2006	The signal light_intensity shall be an input to lights control. Its value shall range fro		
	2008	lights control shall provide an applicable parameter MIN_LIGHT_OFF (which repres		
	2007	The signal headlight shall be provided by lights control. It shall distinguish two val		
	2009	lights control shall provide an applicable parameter MIN_LIGHT_ON (which repre		
	2010	lights control shall provide an applicable parameter HYSTERESIS_TIME_ON, (which		
	2011	lights control shall provide an applicable parameter HYSTERESIS_TIME_OFF, (which		
	201	Functional requirements		
	2013	During light_switch operates in mode AUTO and light_intensity is greater than MIN	4	
	2014	During light_switch operates in mode AUTO and light_intensity is less than MIN_Li	3	
	2015	If light_switch starts operating in mode AUTO or changes to mode AUTO during of	2	
	2016	If light_switch starts operating in mode AUTO or changes to mode AUTO during of	2	
	2017	If light_switch is OFF, then headlight shall immediately be OFF.	3	
	2018	If light_switch is ON, then headlight shall immediately be ON.	3	
- Requirement details (Bottom Right):** A table showing the details of the selected requirement '2017'.

Attribute	Value
ID	2017
Text	If light_switch is OFF, then headlight shall immediately be OFF.
URI	http://vs7.piketec.local:3000/cb/issue/2017
Comment	
Description	--
Modified at	2021-04-09T07:51:57.294

Figure 4 Imported Test Cases and Linked Requirements in TPT

In codebeamer you can categorize your test results, for example, per model or integration level. codebeamer also provides an overview over the entire project including all artefacts and their status, thereby considerably facilitating test project management.

After test cases have been executed, the results can be exported to codebeamer. Results are maintained in the Test Run tracker, as shown in the figure below.

codeBeamer Application Lifecycle Management (ALM)
Evaluation only! Not for production use!

My Start | Projects | Reports | Review Hub | Wiki | Documents | Trackers | SCM Repositories | Baselines | Admin | Trash

Lights Control Introduction > Trackers > Test Runs > Quick Test Run for OFF ON at Apr 16 2021
Run of OFF ON #3047 HEAD / v2

Upstream References TESTCASE-2121 Level 1 req. 2017 req. 2018

Tags: not added yet

Test Results: Quick Test Run for OFF ON at Apr 16 2021

Tracker: Test Runs Parent: Quick Test Run for OFF ON at Apr 16 2021 Test Set: --

Priority: -- Status: FINISHED Result: PASSED

Completed at: Today 15:48 Release: -- Test Configuration: --

Build: -- Submitted by: je Today 15:48 Modified by: je Today 15:48

Assigned to: -- Running Time: 0s Formality: --

Reported bug(s): --

Test Case: [TESTCASE-2121] OFF ON v 30

Step	Action	Expected result	Actual result	Result
#1	Turn Light Switch OFF	Light is OFF	--	--
#2	Turn Light Switch ON	Light is ON	--	--
#3	Turn Light Switch OFF	Light is OFF	--	--

Traceability Show Associations Number of Levels: 2 Do not show redundant items

Figure 5 Test Run View for Test Case after Results Export to codebeamer

Throughout all of the described phases artefacts are linked to one another across both tools, maintaining constant co-traceability.

Release management

For new releases one would import changed requirements into TPT. TPT then highlights all changed requirements since the last release and highlights all artefacts in TPT that are related to the changed requirement. All test cases that are affected by a changed requirement are highlighted, as well. Changes in requirements can also be easily tracked. This makes working on new releases more efficient.

Summary

This white paper highlights the integration of codebeamer and the testing tool TPT, describing the development and requirements-based testing with both tools throughout the release cycles.

First, codebeamer and the testing tool TPT are briefly introduced. Second, the relation of the artefacts, as well as the consistent exchange of information between both tools in the testing process is outlined, stressing the central role of requirements-based testing.

In the following, the development process through multiple release cycles is described. This section lays out the project management, test management and legal requirements of developing safety critical software, identifying codebeamer and the testing tool TPT as particularly suitable full round trip support tools thanks to their integration.