Razorcat Development GmbH

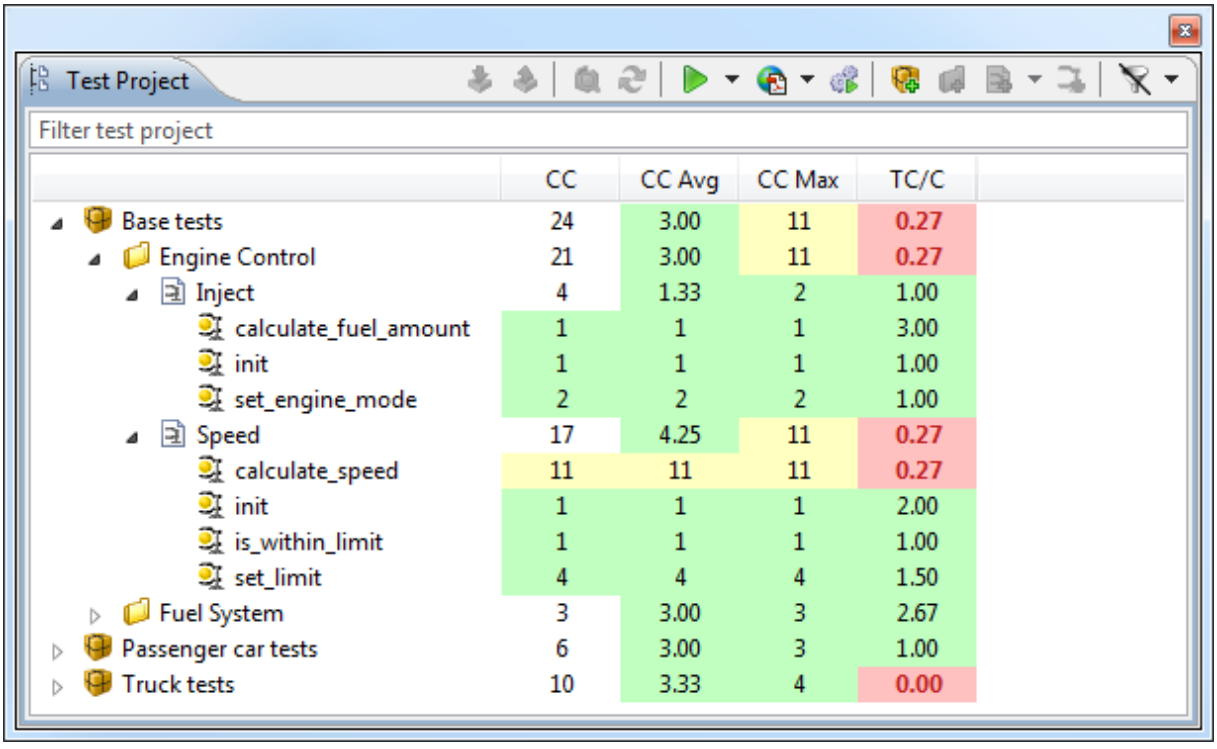
Email: support@razorcat.com

Phone: +49 - 30 - 536 357 0

# New features in TESSY v4.1

## Static analysis and test case quality metrics

Calculation of the McCabe metric (cyclomatic complexity, CC) has been added. The complexity value for each test object will be summarized on module, folder and test collection level using either the sum of all values, the average value or the maximum value.




| | CC | CC Avg | CC Max | TC/C |
|---|---|---|---|---|
| Base tests | 24 | 3.00 | 11 | 0.27 |
| Engine Control | 21 | 3.00 | 11 | 0.27 |
| Inject | 4 | 1.33 | 2 | 1.00 |
| calculate_fuel_amount | 1 | 1 | 1 | 3.00 |
| init | 1 | 1 | 1 | 1.00 |
| set_engine_mode | 2 | 2 | 2 | 1.00 |
| Speed | 17 | 4.25 | 11 | 0.27 |
| calculate_speed | 11 | 11 | 11 | 0.27 |
| init | 1 | 1 | 1 | 2.00 |
| is_within_limit | 1 | 1 | 1 | 1.00 |
| set_limit | 4 | 4 | 4 | 1.50 |
| Fuel System | 3 | 3.00 | 3 | 2.67 |
| Passenger car tests | 6 | 3.00 | 3 | 1.00 |
| Truck tests | 10 | 3.33 | 4 | 0.00 |

As a derived measure based on the complexity, the TC/C ratio defines the number of test cases necessary to reach 100% branch coverage. A value greater than 1 indicates that at least a minimum number of test cases has been defined.

Another new measure for the quality of test cases, the result significance (RS), reveals weak test cases: It verifies that each test case has at least some expected results, checks the call trace or uses evaluation macros.

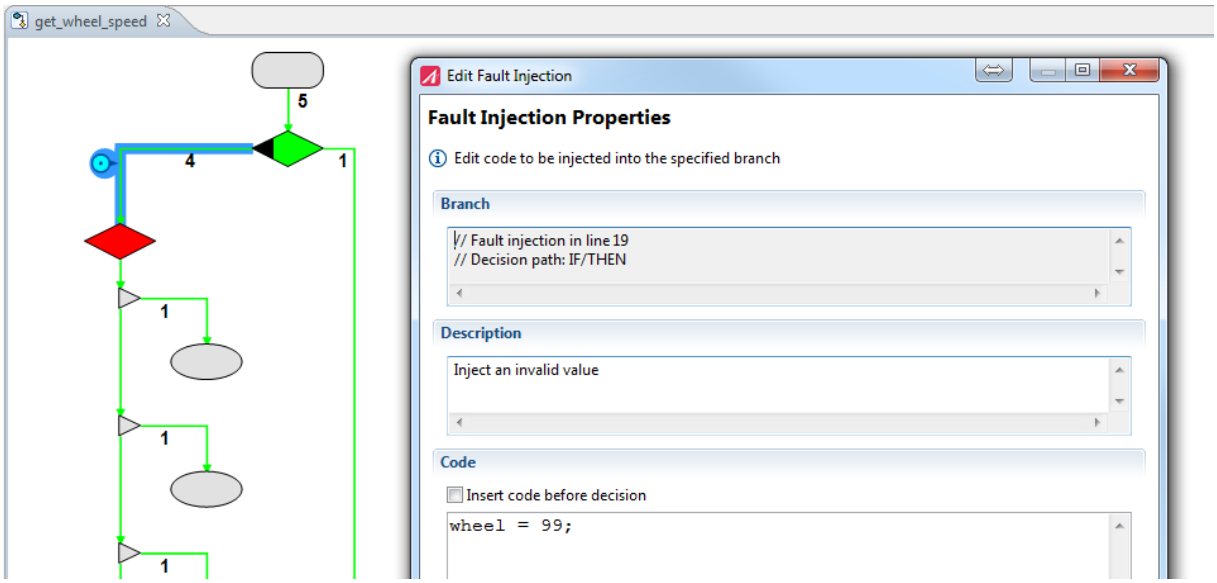
## Fault injection

The new feature for automated fault injection on source code level overcomes problems during test implementation caused by programming paradigms that are widely used within safety critical software engineering:

- Defensive programming
- Read after Write
- Endless loops

The necessary code coverage of 100% could not be reached with normal test cases so that in practice additional testing code or code replacements using compiler macros were implemented in the past.

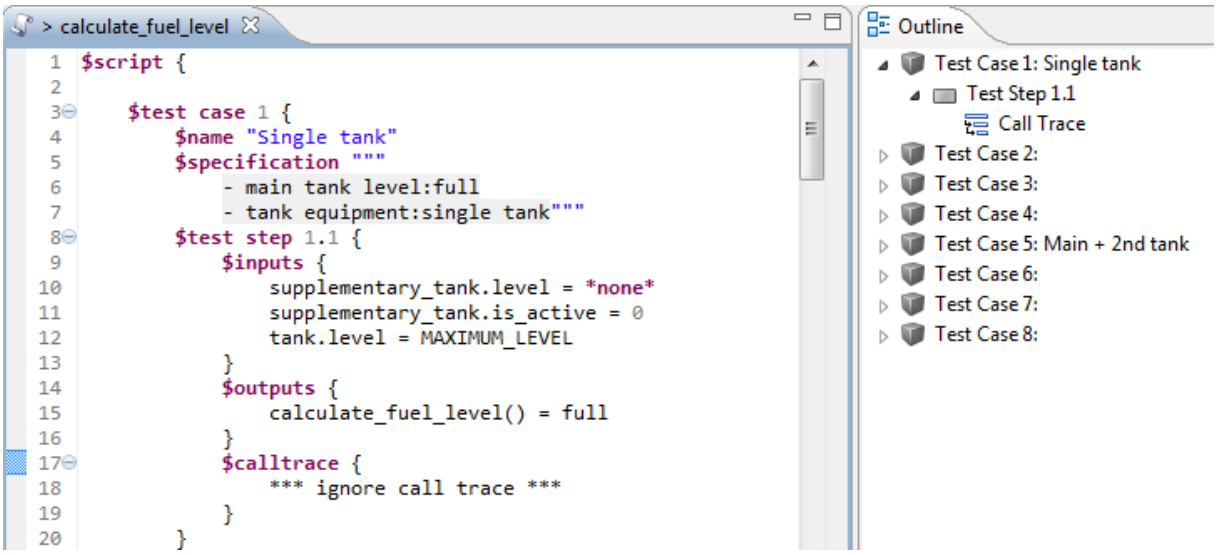
Now TESSY provides automated fault injection without affecting the original source code. The fault injection will be placed into the flow graph of the tested function and it will be active only for one or several test cases where it is assigned to. Such fault injection test cases will be marked and documented within the test report.




Fault injections are created based on unreached branches of the function flow graph, they can be applied without any change to the source code and they will be effective at the desired location even after source code changes when doing regression testing.

## Script perspective

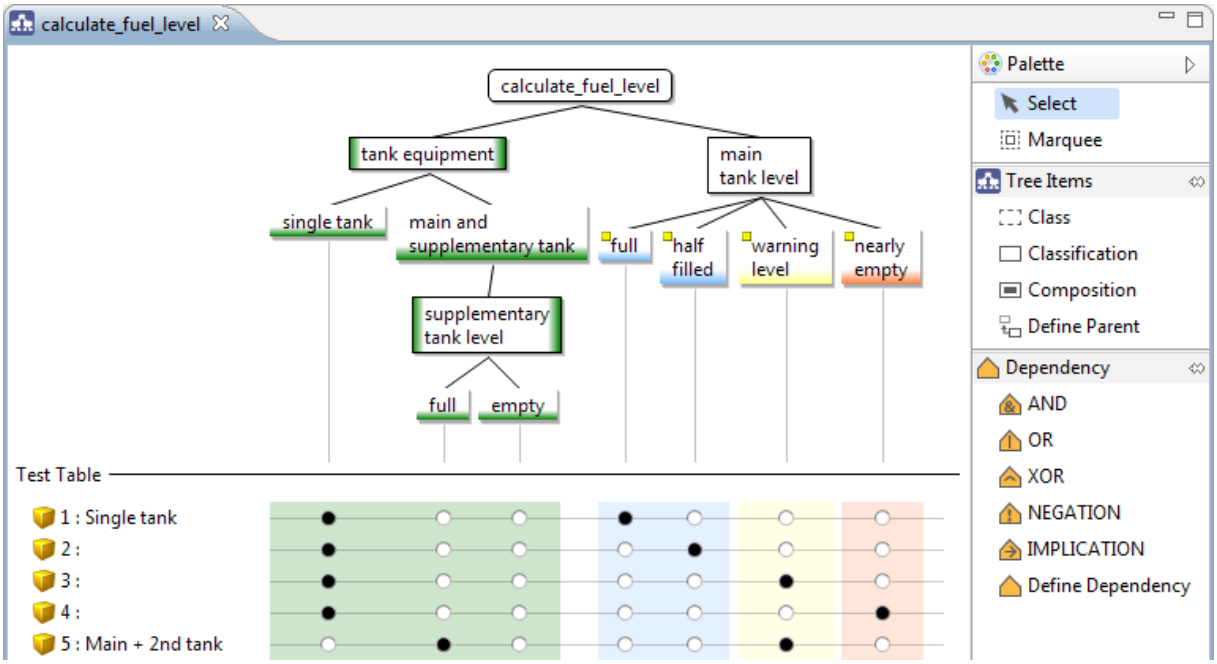
The new script perspective allows to edit tests within an ASCII editor using a dedicated test scripting language. All test data can be converted from the script format to the TESSY internal format and vice versa. The ASCII format can also be used as complementary backup format besides the hitherto TMB file format.




The script editor provides syntax highlighting, auto completion, formatting, validation and an outline view.

## New classification tree editor (CTE)

The new implementation of CTE is a full featured eclipse-based editor integrated into TESSY which enhances the design of test cases and assignment of test data to tree nodes.

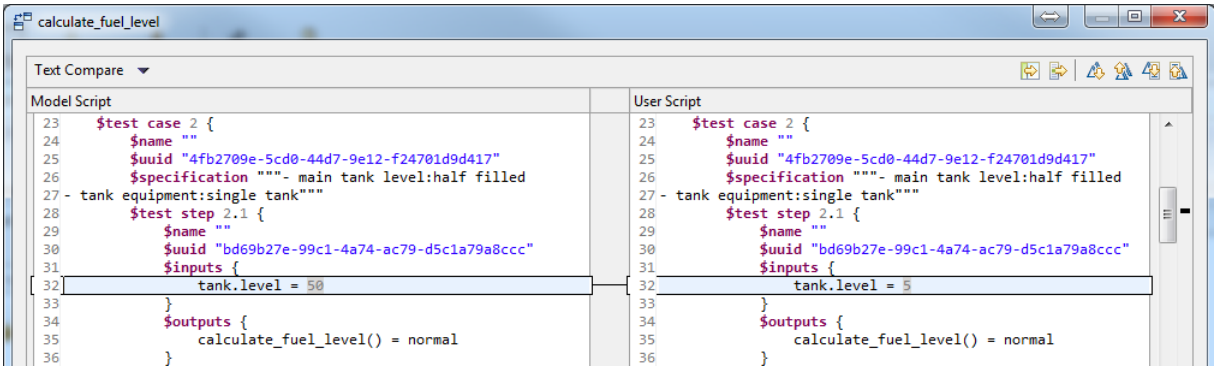



Legacy CTE trees will be updated and converted to the new CTEX file format automatically when they are opened for editing.

## Review of test changes

The new test scripting language provides the means to store the test contents for each test object as ASCII file beside the binary module archive (TMB file). This allows easy versioning

of tests and comparison of arbitrary versions using a version control system. For reviews of tests, changes can easily be found using standard ASCII diff tools.




## Arithmetic expressions as test data

The TDE allows to enter arithmetic expressions as test data values. The resulting value of an expression will be calculated and updated on code changes. The following operators are supported: Addition, subtraction, multiplication, division, shift, binary or/and. The operands can be numbers, defines and enum constants.

## Stimulating and measuring hardware signals

In order to enable hardware I/O stimulation and measurement during unit testing, TESSY provides a hardware adapter interface allowing control of external measurement hardware. This hardware device has to implement a configuration interface as well as reading and writing methods for hardware signal data.

During module analysis, TESSY reads the configuration of the hardware device in order to determine the available interface (i.e. the available I/O signals).This list of signals (including passing directions) will appear within the interface of the TESSY module (for each test object). The signals may be edited within the test data editor (TDE) as any other normal test object variable.