Tessy V2.6 Features

Enhanced Coverage Visualization and Reporting

Tessy provides now graphical visualization and reporting for coverage measures in flowchart form.

You may interactively display the relation of branches in the graph to source code. For each decision in the graph, you may interactively display the Modified Condition / Decision Coverage (MC/DC) or Multiple Condition Coverage (MCC). It is also possible to display the branches taken by one or several test cases.

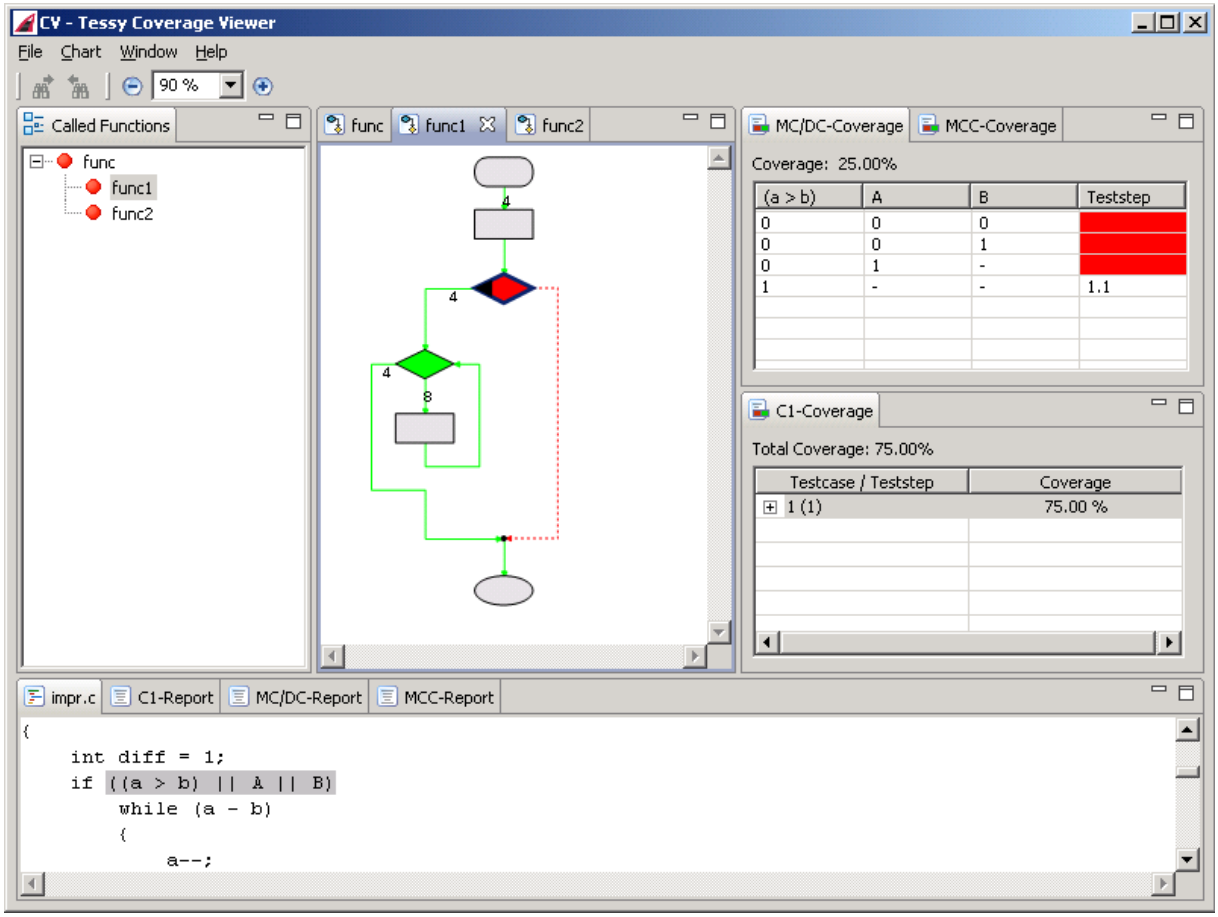

Fig: Coverage Viewer

Besides the interactive investigation of the Code Coverage, summary reports are available. These can be printed or stored in files.

Automatic Test Data Generation

A context menu in the Test Data Editor (TDE) allows for automatic generation of test data. The test data manifests in the test steps of a single test case. The test steps are generated by combination of values for the variables in the interface of the test object. The user may simply specify to use minimum and maximum values; alternatively, the user may specify a list of values or a range of values to be used for combination. Tessy calculates the number of test steps resulting from the current settings. This allows the user to avoid inflation of test steps by adjusting the settings. Eventually, the user may append the generated test steps to existing test steps of this test case or the user may create a test case consisting purely of generated test data.

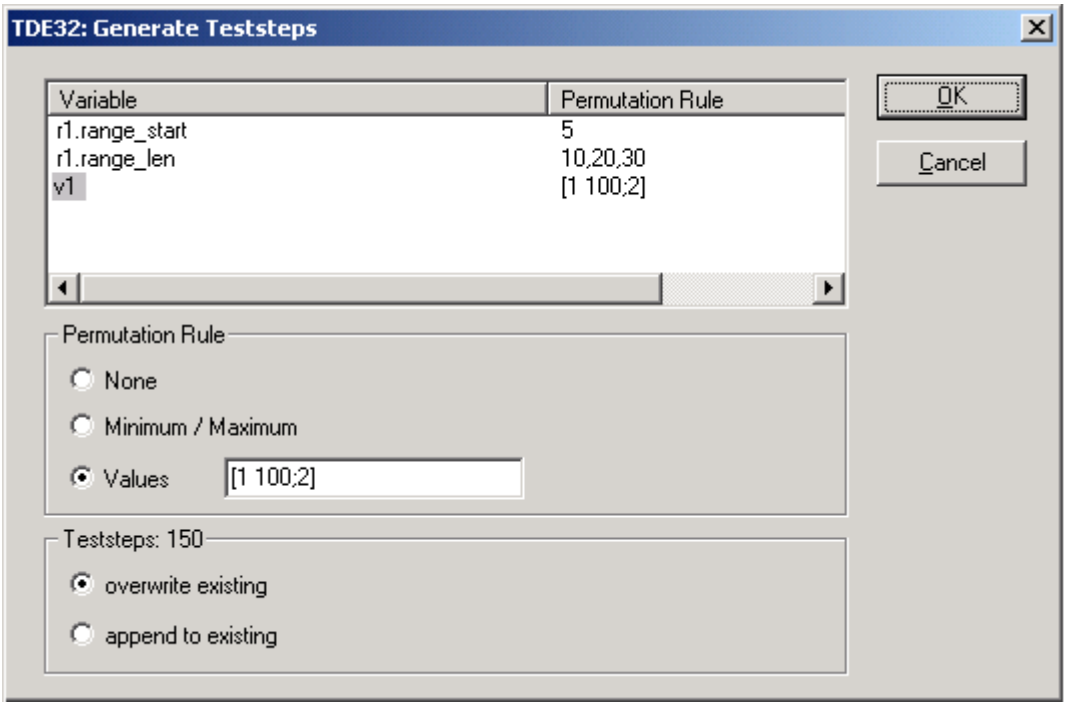

Fig: A range from 1 to 100 with step width 2 is specified for the variable v1

Graphical View of Variable Values

A new test data editor allows graphically visualizing test data for interface variables as signal data row. This is available either for data rows of variables over all test cases or over all test steps of a single test case.

Graphical result plots of interface variables are available without additional software (e.g. Matlab).

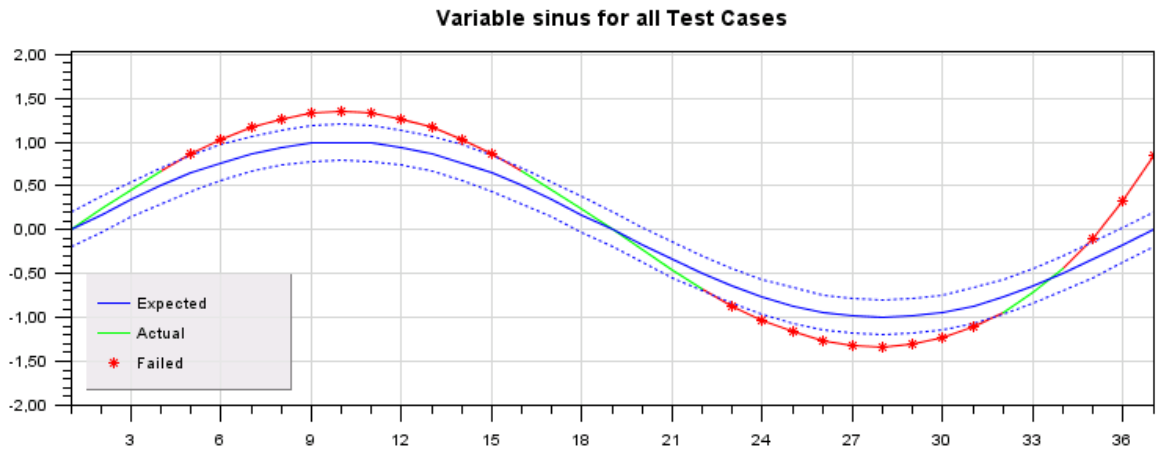

Fig: Graphical test result evaluation

The blue line depicts the exact expected result. A deviation of +/- 0.2 is accepted in the test at hand (dotted blue lines). Results within the accepted deviation are displayed in green (i.e. passed tests); results outside the accepted deviation are displayed in red (i.e. failed tests).

C++ Support

Tessy now supports testing of C++ modules for certain compilers supporting the C++ language. Test data entry and result evaluation will be accomplished in the Usercode of Tessy.

Using Symbolic Constants as Test Data

Tessy allows to use #defines of constant values as symbolic test data. E.g., if the user's source code contains

```
#define WHITE   0
#define RED     1
#define GREEN   2
#define BLUE    3
```

up to now, "2" has to be entered in Tessy's Test Data Editor (TDE) to denote "GREEN". In Tessy V2.6, "GREEN" can be entered directly as test data.

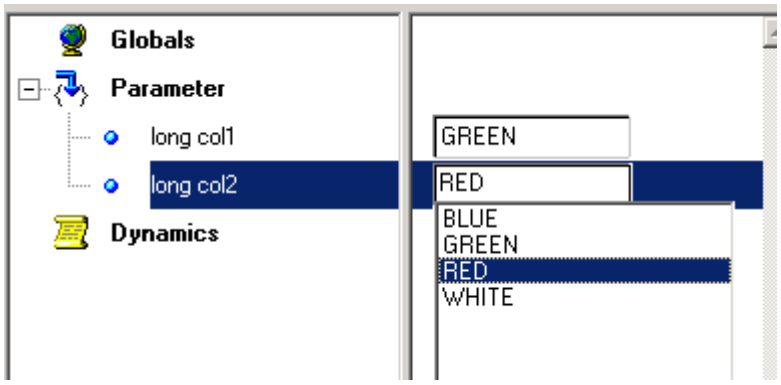

Fig: Defines can be entered in the TDE

This eases the definition of the test cases and help to understand their semantics. Furthermore, if the definition of “GREEN” should change during development, e.g. to

```
#define GREEN          20
```

the value for GREEN in the test data will change automatically, too.

Definitions are also available in the CTE.

Bit Check

Individual bits of a variable can be checked during the evaluation for a test case result. It is be possible to mask-out bits of an expected value in the Test Data Editor. Fore this, a new format for expected values is introduced, e.g.

```
0b000xxxx1111000
```

where “0b” is the format specifier, “0” denotes an unset bit, “1” denotes a set bit, and “x” denotes a don’t care bit in the expected result.

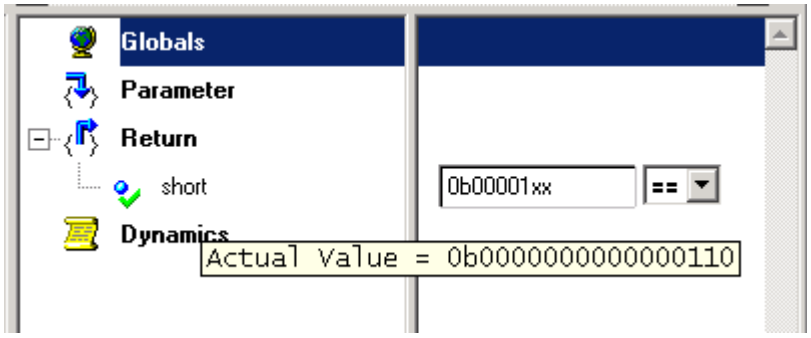

Fig: For the evaluation of this test result, the last two bits were not relevant

Value for the Return Value of Functions in the CTE

If a function is not void, but has a return value, it is now possible to associate a classification with the return value and to specify a value for the expected result in the underlying classes. This value is automatically used in the test cases exported to Tessy. This is similar to what was already available for normal output variables.

Interface Database Optimization

The improved Interface Database (IDB) allows restoring the original passing directions after any changes to the interface settings.

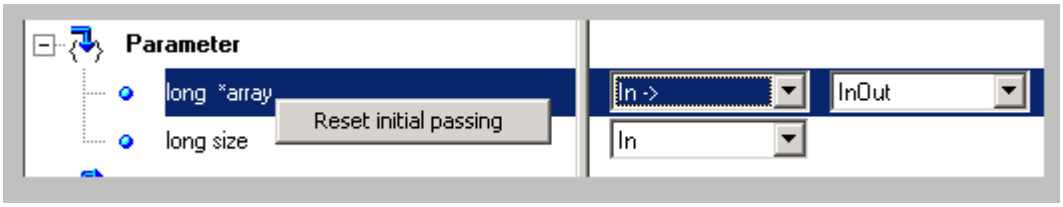

Fig: The Test Interface Editor (TIE) allows resetting the passing direction

Also new interface analysis findings of Tessy due to source code changes are reflected within the original passing directions settings after re-use operations.

Variables and functions used or referenced within initializing statements are available within the Interface Data Base (IDB).

The passing direction of module global variables, which are only used by module local functions, are set to “Irrelevant” if the local function gets stubbed.

Optimization of the IDB provides faster response times for larger interfaces.

Tool Qualification Package

A tool qualification package for use within certification activities according to standards like DO-178B, IEC 61508 or Automotive SPICE will be available. Customer specific adaptations or consulting services will also be offered.

Improved Tessy Environment Editor

The Tessy Environment Editor is optimized to allow simplified access to the most frequently used settings. An Expert Mode is introduced to allow access to more esoteric settings.

Fig: The TEE now features an Expert Mode

New Installer

A new install tool is used.

Support for Windows Vista

Tessy now runs under Windows Vista.

