Atos

User Manual

Edition January 2014

Copyright © Atos IT Solutions and Services GmbH 2014

Microsoft, MS, MS-DOS and Windows are trademarks of Microsoft Corporation.

The reproduction, transmission, translation or exploitation of this document or its content is not permitted without express written authority. Offenders will be liable for damages. All rights reserved, including rights created by patent grant or registration of a utility model or design.

Right of technical modification reserved.

Contents

1 Overview

Managing the test process involves a large number of engineering disciplines like project management, risk management, high and low level design, development, and metrics collection. Normally testers manage their test cases with Word tables or Excel sheets. The main disadvantages are that test cases cannot be maintained easily, groups of test cases cannot be divided flexibly, no statistics can be provided automatically, etc.

TEMPPO (**T**est **E**xecution, **M**anagement, **P**lanning and re**P**orting and **O**rganizer) was developed to make the test process and test management easier and to provide methods for test case design and development as well as test case execution and integration of test automation tools.

TEMPPO consists of

- **TEMPPO Test Manager**: the test management application
- **TEMPPO Requirement Manager**: the requirement management application
- **TEMPPO Designer**: the application for test case and test data generation
- **TEMPPO Administrator**: the application for meta data maintenance
- **TEMPPO Manager**: the application for database migration

The main parts of **TEMPPO Test Manager** are:

- Editing and administrating automated and manual test cases including version control
- Integration of automated test case execution with WinRunner, SilkTest, TestPartner, QuickTest Professional, Ranorex or any other tool that supports a command line interface
- Flexible manual test case execution (even with unplugged notebooks)
- Professional analysis, statistics and reporting

This manual described the way of working with TEMPPO Test Manager.

2 Introduction

When we talk about testing, we consider a **test case** to be the basic unit of discussion. A test case is defined by:

1. a descriptive name (e.g. "Create new customer account")
2. a single, well delineated **test goal** (e.g. "create a new customer account in the data base")
3. a sequence of **test steps** (e.g. "1. Log in, 2. Open customer creation dialog, ..."),
   consisting of instruction, input data and expected result.

Additionally, **attributes** may be defined for test cases, like

1. test situation (does it test a normal operation or an error condition)
2. state (is the test case still in work or ready for execution)
3. priority (how important is it to execute this test case before the product is released)
4. type (manual, i.e. test steps have to be performed by a human operator, or automated)

The Test Process

All testing activities deal with planning, designing, implementing, executing and managing a sometimes quite big amount of test cases. TEMPPO supports a well-defined approach to these activities. We can distinguish five phases in the test process:

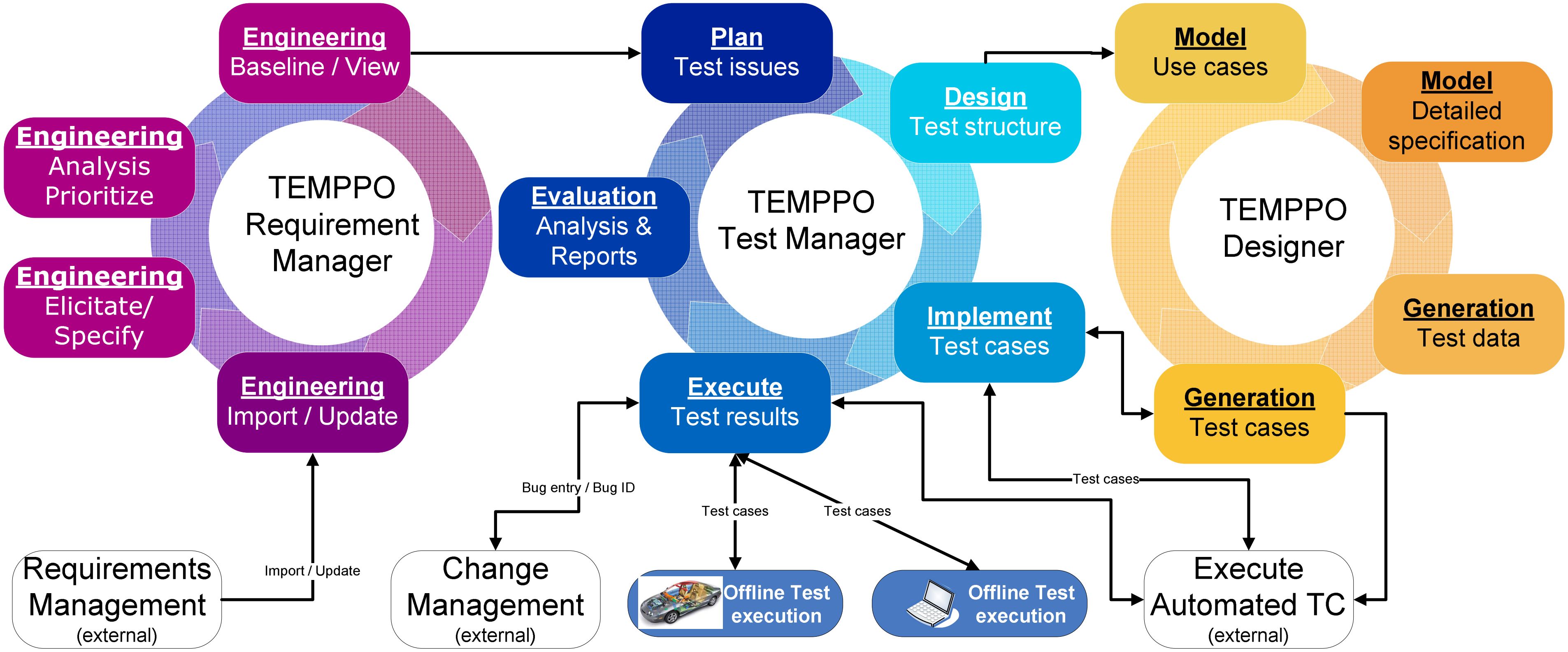


Figure 1 - Test Process with TEMPPO

Test Planning. This phase covers mainly writing the test plan and a project plan, typically in a planning tool like MS project. A test plan template can be found in the TEMPPO installation directory under /doc and it takes into account the following issues:

- test requirements (baselined requirements in TEMPPO Requirement Manager)
- test management (responsibilities, planning and organization, CM, etc.)
- preparations for the test (e.g. setting up the test infrastructure, creating test aids, conventions, etc.)
- tests to be performed (including test goals, end-of-test criteria, test procedures, sequences, etc.)
- test cases
- test documentation
- error reporting procedure

TEMPPO supports the issues test cases (planning, design, implementation, execution) and test documentation (evaluation), i.e. you can refer to TEMPPO in your test plan for these topics.

Although your testing activities just started in the phase of test planning, you might already anticipate the situation as it will be at the beginning of the test case execution or evaluation phase: by that time you will have a lot of test cases. However you have to consider now, how test cases should be designed, to make it easy to

1. select a specific subset of your test cases for test execution
2. verify the test goals or end-of-test criteria

For this purpose a test case in TEMPPO can have several attributes, which enable flexible filtering and the creation of analysis charts and reports under specific aspects. And you should check during the test design phase whether the attributes provided by default are sufficient. Otherwise, you would define additional ones such that your needs will be fulfilled.

Test design. Test design is either done with TEMPPO Designer (available as Add-on, please refer to chapter 3.3.20.2.9) or with TEMPPO Test Manager. With **TEMPPO Test Manager** you will think about organizing your (future) test cases in some way. Since a simple list of dozens of items is not easy to survey, a hierarchical grouping of test cases seems adequate. TEMPPO supplies the concept of **test packages**, which may contain test cases or, for building up hierarchical structures, again test packages. The whole tree of test packages and test cases is called **test structure**. According to the various implementation levels during a development process – from coding the components and assembling them (integration) to completing the system – the test process has various **test levels**, where you may test the stand-alone components (*module test*), "the growing application" (*integration test*) and finally the complete system (*system test*). Furthermore your customer might execute an *acceptance test* for verifying whether the delivered product agrees with the ordered features. You have to decide, how many test structures you want to build: one for all test levels or different ones for different test levels. Especially for the *system test*, it is highly recommended to build a test structure that closely resembles the structure of chapters in your requirement specification. For that sake, TEMPPO supports the automatic creation of test structures out of RTF-formatted text documents.

Now that you have prepared everything to organize and execute test cases, you are ready to start with their design. During test case design only test case names, test goals and the links to the requirements are fixed. This is useful, because you will quickly get an overview of the number of test cases and their distribution to test packages. And it is still early enough to set corrective measures if e.g. the number of planned test cases turns out to be way too high (or even too low). For a high rate of error finding it’s suggested to make use of well-tried test case design methods like Equivalence Class Partitioning or Boundary Value Analysis (for details refer to /2/).

Test Case Implementation. In this phase, three main tasks have to be accomplished: (i) writing manual test cases (test step definition), (ii) writing automated test cases (test script programming), (iii) implementing test frameworks. Only the first task can be covered by a test management tool like TEMPPO. For test script programming, several tools exist on the market. The role of TEMPPO in this context is to provide a connection between a test case designed within a test package and the implementation of its test steps as a test script on the file system. The implementation of test frameworks is completely outside the scope of TEMPPO.

Test Case Execution. Now you can reap the benefits of hard test case implementation work and you are ready to create collections of test cases for test execution (so called **test suites**) by means of combining the attributes defined above. So the appropriate test cases can be filtered from the pool of all available ones. Therefore the number of test cases of a test suite is always a subset of those of the corresponding test structure. For each test structure you can create as many test suites as you like. E.g. you have the possibility to separate test execution tasks by creating special test suites for each tester. Accordingly it is important to write a good test plan, especially to plan attributes, which are used here (see planning phase). If you recognize in the execution phase, that there is an attribute missing, this will cost some time to add it for *each* test case.

Evaluation . This is the final phase of your test cycle. After executing the tests and recording their results, you will analyze them and provide reports, which illustrate your testing activities and the quality of the tested product. For this reason TEMPPO offers features, which allow the creation of analysis charts and detailed, customizable reports of test structures and test suites.

3 Description of Use

3.1 The Test Process Phases

As stated in the introduction chapter, the whole process of testing software products can be split up into five phases:

1. Test Planning
2. Test Design
3. Test Case Implementation
4. Test Case Execution
5. Evaluation

The following chapters describe in detail, how the tasks of these phases can be solved more efficiently by using the TEMPPO test management tool. During all chapters a pseudo system test for the MS Windows Notepad application will illustrate the usage of TEMPPO more clearly.

3.1.1 Test Planning

The definition phase covers the tasks of

- Writing the Test Plan
- Writing the Project Plan
- Defining Test Case Attributes
- Baselining / Importing requirements

3.1.1.1 Writing the Test Plan

The test plan is a process document and writing is outside the scope of TEMPPO. However, it’s the base document for all test topics during the development process and covers subjects like test requirements, test management, test documentation, test goals, end-of-test criteria, etc. Its contents are obligatory for all members of the test team.

3.1.1.2 Writing the Project Plan

As an output of the test plan a Project Plan (e.g. MS Project) is written that contains tasks for all test activities. Input for the plan is given by the

- general budget for testing activities
- estimated number of test cases to be developed
- number of resources, milestones
- estimated effort of executing a test case
- bug analysis
- regression testing (testing fixed bugs) effort

3.1.1.3 Defining Test Case Attributes

After the implementation phase you will have a lot of test cases (even in small projects) and it's very likely, that you want to have just a few of your test cases bundled for test execution, because

- you don't have the time to execute all of them and therefore you decide to execute the high priority ones only.
- there are several testers, each of which should execute several test cases.

Furthermore we could consider, that you want to get a survey of your test cases or test results under certain criteria. For example it could be interesting for you,

- how many test cases have the implementation state ready (which means, they can be executed).
- how many high priority test cases are failed.

TEMPPO offers features for both situations, namely the filtering of test cases for test execution (see 3.1.4.1) and the creation of analysis charts (see 3.1.5.1). But for being able to use these features, it is indispensable to define attributes. The following test case attributes are predefined in TEMPPO:

- Type (Manual, Automated)
- Situation (Regular, Error)
- Priority (Top, High, Medium, Low)
- State (Designed, In Work, Ready for Review, Ready, Approved)
- Test case type (Verification, Validation)
- Test Level (Module Test, Integration Test, System Test, Acceptance Test, Regression Test)

In most cases the predefined attributes will fulfill your needs and you just have to decide, which ones to use (defined in the test plan). If you cannot come along with them, you have the possibility to define your own ones, so called user defined attributes. For detailed information on how this is done in TEMPPO, see 5.4.4.

3.1.1.4 Baseline / Import Requirements

SiTEMPPO Requirement Manager is used for requirement engineering or for importing and updating your requirements in TEMPPO's database from external tools like DOORS or RequisitePro.

If the requirements are ready for test case creation the requirements are baselined using also the view concept. The requirement engineer defines a filter from requirement attribute (e.g. All requirements that are implemented and of high priority) and freezes that requirements to a baseline.

On the other hand the requirement engineer imports and later updates the requirements from the external tools.

For details refer to **TEMPPO Requirement Manager's User Manual.**

3.1.2 Test Design

Test Design covers the tasks of

- Working with Projects
- Building the Test Structure
- Extending the Test Structure (adding more test packages)
- Designing Test Cases (test case design)

3.1.2.1 Working with Projects

A project is the starting point in TEMPPO and contains one or more test structures. You can either start by creating a new project or opening an existing one. Deleting a project can only be done with the TEMPPO Administrator (see 5).

If you are an advanced user, you can skip this step and directly choose a previously opened test structure in the `Test Structure` menu (see 3.1.2.2.3).

3.1.2.1.1 Creating a Project

A new project can only be created in TEMPPO Administrator. Log on to TEMPPO Administrator and choose the tab Projects. Then press the button "`New`". Figure 2 is displayed.

Project properties contains 4 tabs:

-General

-Roles

-Test Case Template

-Test Result Template

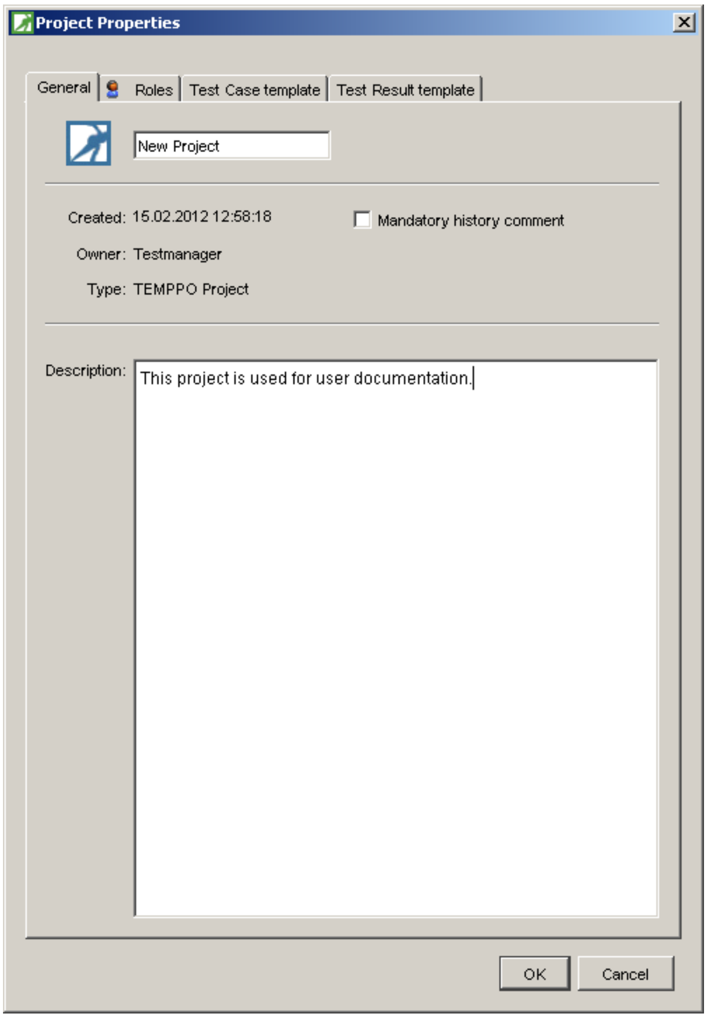
In the tab **"General"** user specifies a unique name, description and the possibility to determine if the testers have to add a mandatory comment for each save operation of a test case.

In the tab **"Roles"** the test manager configures the roles for each tester. If a user has no role in a project, he won't see it.

In the tab **"Test Case Template"** the test manager appoints defines a basic test case. He can configure standard attributes, define mandatory attributes or determines if a test case has to be at least one link to a requirement. If these conditions are not fulfilled the test case cannot be saved. For details see chapter 5.5.1.

In the tab **"Test Result Template"** the test manager configures result attributes for test case. If a tester records a result and a test result template is activated, he has to add a value for the mandatory attribute. For details see chapter 3.3.3.2.

After pressing the button "OK", the user is asked if he wants to assign immediately meta data. If the user presses "Yes", Figure 341 is shown.



New Project

Created: 15.02.2012 12:58:18

Owner: Testmanager

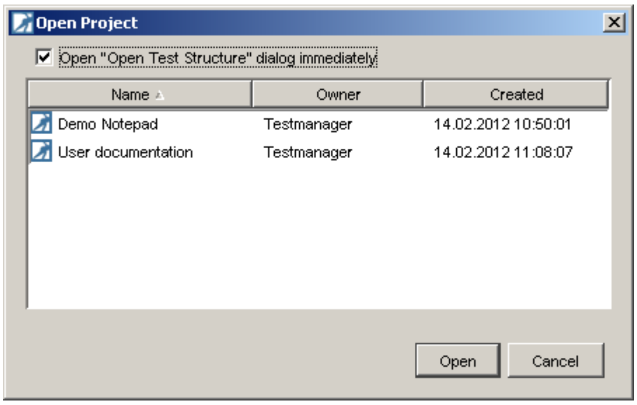
Type: TEMPPO Project

☐ Mandatory history comment

Description: This project is used for user documentation.

Figure 2 - Project Properties – TEMPPO Administrator

3.1.2.1.2Opening an Existing Project



☑ Open "Open Test Structure" dialog immediately

Name	Owner	Created
Demo Notepad	Testmanager	14.02.2012 10:50:01
User documentation	Testmanager	14.02.2012 11:08:07

Figure 3 - Open Project

When activating the menu item **Project > Open,** the window **Open Project** (Figure 3) with project **name**, **owner** and **creation date** is displayed.

Exactly one project can be selected and is opened by clicking the **OK** button. The name of the project is shown in the title bar of the main window (see Figure 4)

Cancel closes the window and no project is opened.

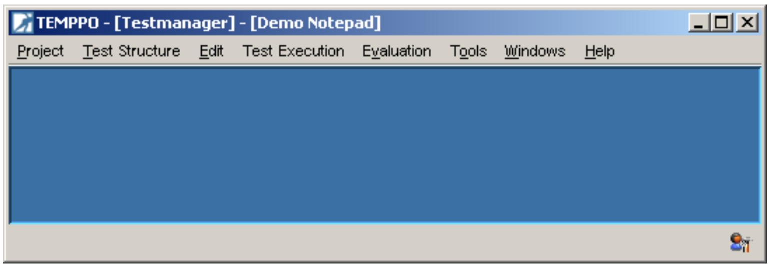

TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help

Figure 4 - Main Window (project opened)

3.1.2.2 Building the Test Structure

A test structure contains test packages and test cases, which can refer to requirements. This can be established in 2 ways (and/or):

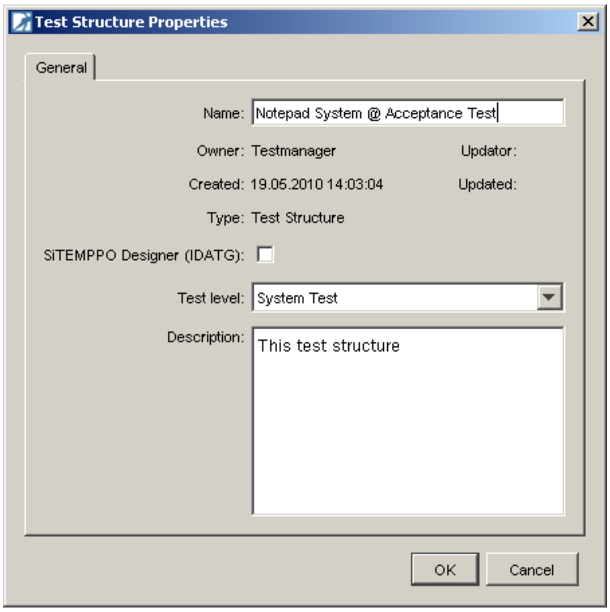
- Referring to (chapters) in documents
- Referring to requirement numbers or names

3.1.2.2.1 Creating a test structure

TEMPPO allows opening only one test structure at a time, which means, that a test structure can only be created, if no other one is already open. Furthermore test structures are related to projects, so the project, which shall own the test structure to be created, must be open too.

3.1.2.2.1.1 Creating an empty test structure

When activating **Test Structure > New… > Empty Test Structure** the window shown in Figure 5 is displayed. You may enter a **name** and a **description**. Also a **test level** has to be related to the new test structure.



Name: Notepad System @ Acceptance Test

Owner: Testmanager

Updator:

Created: 19.05.2010 14:03:04

Updated:

Type: Test Structure

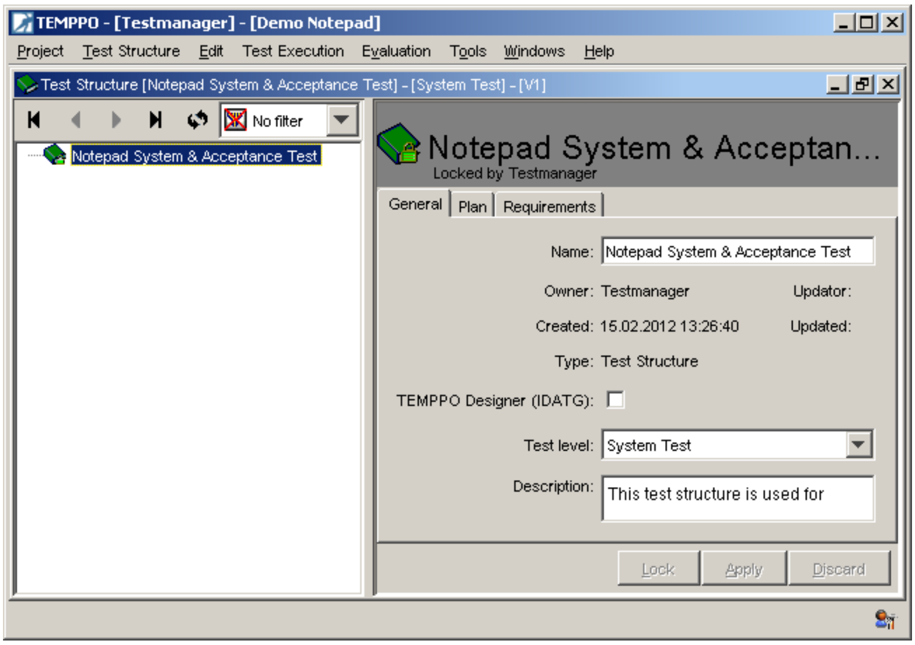
SiTEMPPO Designer (IDATG): □

Test level: System Test

Description: This test structure

Figure 5 - Test Structure Properties

By clicking **OK**, Figure 6 is shown. The test structure is opened in the main window. In the title bar of the test structure its name, test level and version are displayed. In the left pane an empty tree is displayed and in the right part the test structure properties are documented.


TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Structure [Notepad System & Acceptance Test] - [System Test] - [V1]
No filter
Notepad System & Acceptance Test
Notepad System & Acceptan...
Locked by Testmanager
General
Plan
Requirements
Lock
Apply
Discard

Name: Notepad System & Acceptance Test

Owner: Testmanager

Updator:

Created: 15.02.2012 13:26:40

Updated:

Type: Test Structure

TEMPPO Designer (IDATG): □

Test level: System Test

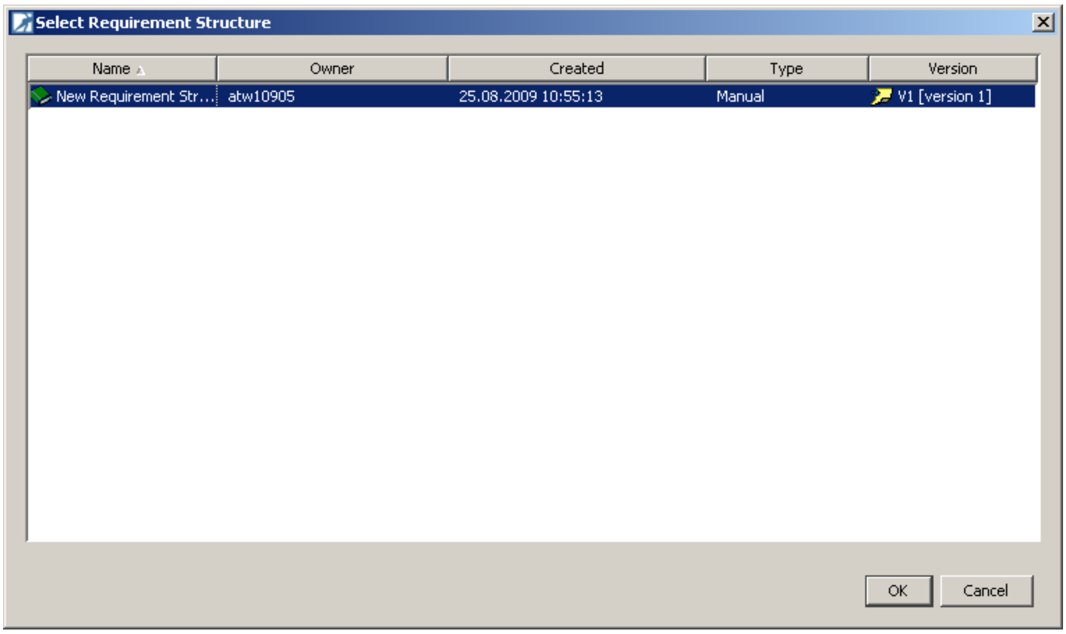
Description: This test structure is used for

Figure 6 - Empty Test Structure (General)

3.1.2.2.1.2 Creating a test structure based on a requirement structure

This feature allows you to create the same structure like in a requirement structure. For each requirement a test package and a dummy test case is created plus automatic link to the requirement.

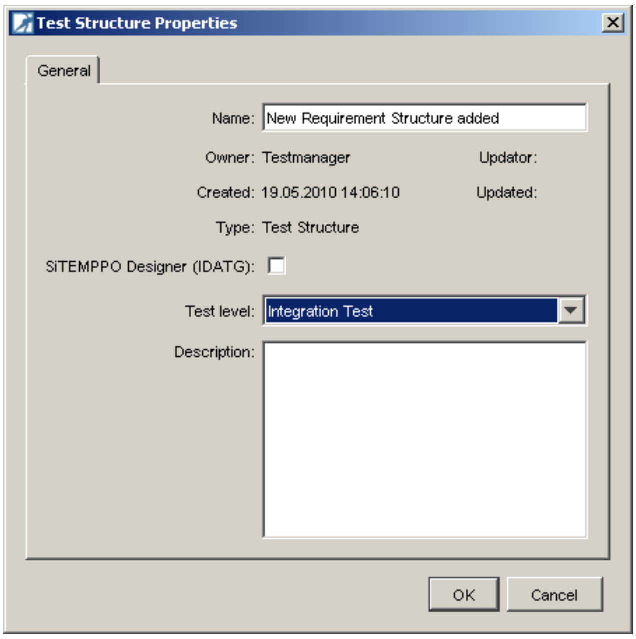
When activating **Test Structure > New... > Based on a requirement structure** the window shown in is displayed. All requirement structures are shown which are assigned to the opened project and checked in (see also Figure 7).



Name	Owner	Created	Type	Version
New Requirement Str...	atw10905	25.08.2009 10:55:13	Manual	V1 [version 1]

Figure 7 – Select Requirement Structure

After selecting one requirement structure another window is shown, see Figure 8. You may enter a name and description. Also a test level has to be related to the new test structure.



Name: New Requirement Structure added

Owner: Testmanager

Updator:

Created: 19.05.2010 14:06:10

Updated:

Type: Test Structure

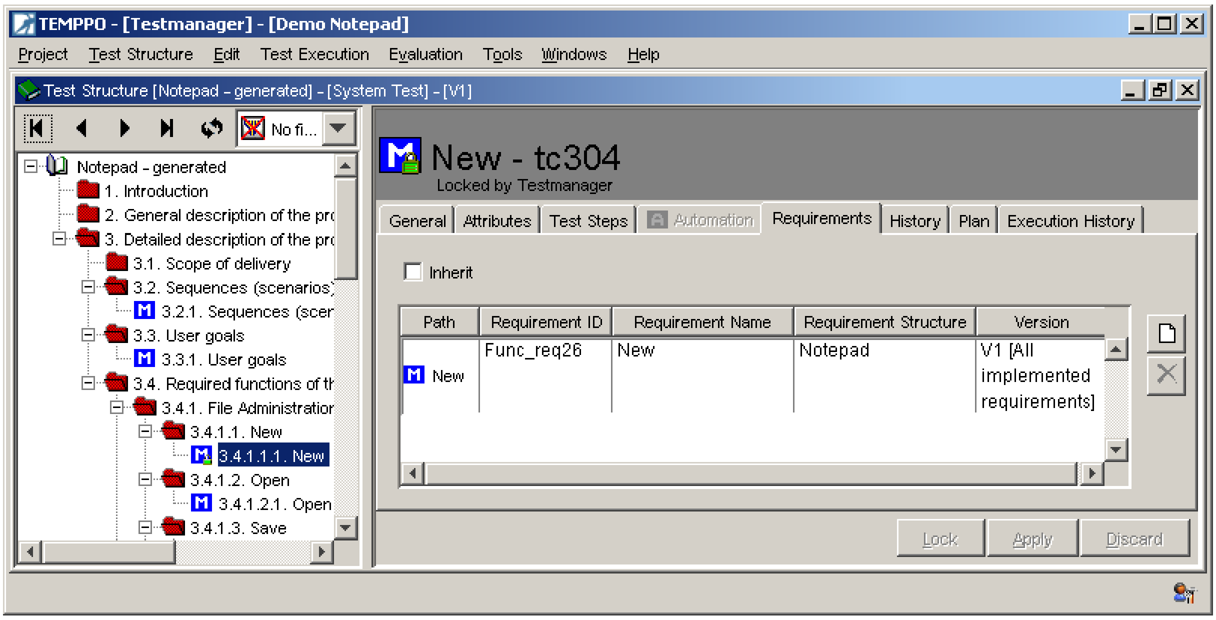
SiTEMPPO Designer (IDATG): □

Test level: Integration Test

Description:

Figure 8 – Test Structure Properties

By clicking **OK**, the test structure is created based on the requirement structure: For each requirement, a test package and a test case is created. Both have a reference to the requirement, described in 3.1.2.3.4. The test structure is opened in the main window, see Figure 9.



□ Inherit

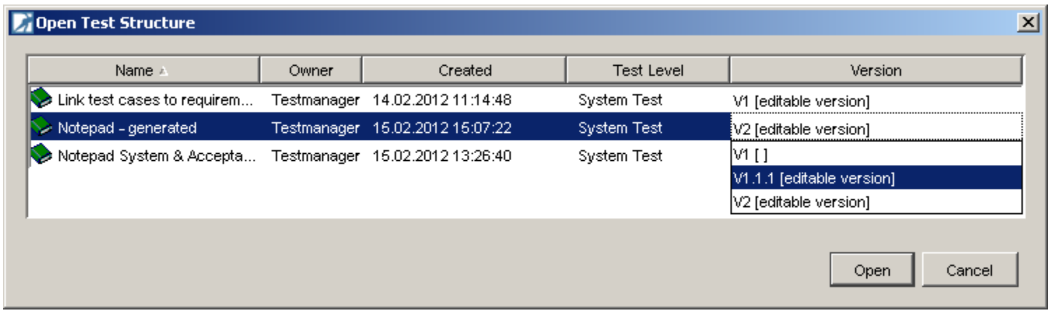
Path	Requirement ID	Requirement Name	Requirement Structure	Version
New	Func_req26	New	Notepad	V1 [All implemented requirements]

Figure 9 – Generated Test Structure

3.1.2.2.2Opening a Test Structure

Using **Test Structure > Open…** will display the dialog in Figure 10. This menu item is only enabled, if a project is opened. Exactly one item can be selected and will be opened after pressing the button `Open`.

Additionally a test structure version (see 3.3.10) can be selected, if a non latest **version** should be loaded.



Name	Owner	Created	Test Level	Version
Link test cases to requirem...	Testmanager	14.02.2012 11:14:48	System Test	V1 [editable version]
Notepad - generated	Testmanager	15.02.2012 15:07:22	System Test	V2 [editable version]
Notepad System & Accepta...	Testmanager	15.02.2012 13:26:40	System Test	

Figure 10 - Open Test Structure

3.1.2.2.3Reopen a test structure

A faster way to open a test structure (and the according project as well) is to use **Test Structure > Reopen**, which displays the 4 last opened test structures. It will only be enabled, if no project is open.

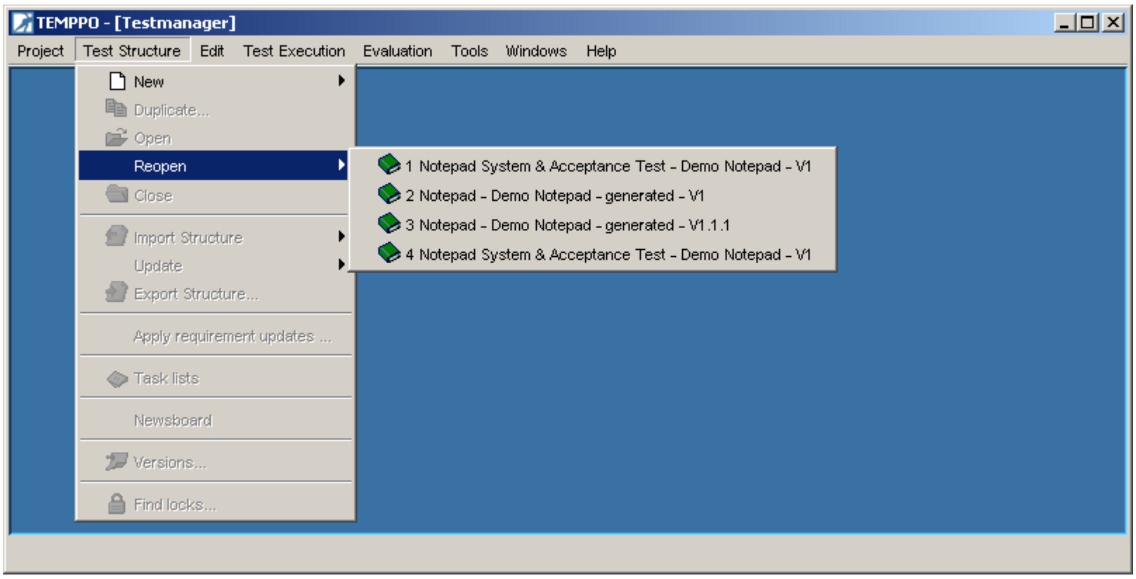

TEMPPO - [Testmanager]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
New
Duplicate...
Open
Reopen
Close
Import Structure
Update
Export Structure...
Apply requirement updates ...
Task lists
Newsboard
Versions...
Find locks...
1 Notepad System & Acceptance Test - Demo Notepad - V1
2 Notepad - Demo Notepad - generated - V1
3 Notepad - Demo Notepad - generated - V1.1.1
4 Notepad System & Acceptance Test - Demo Notepad - V1

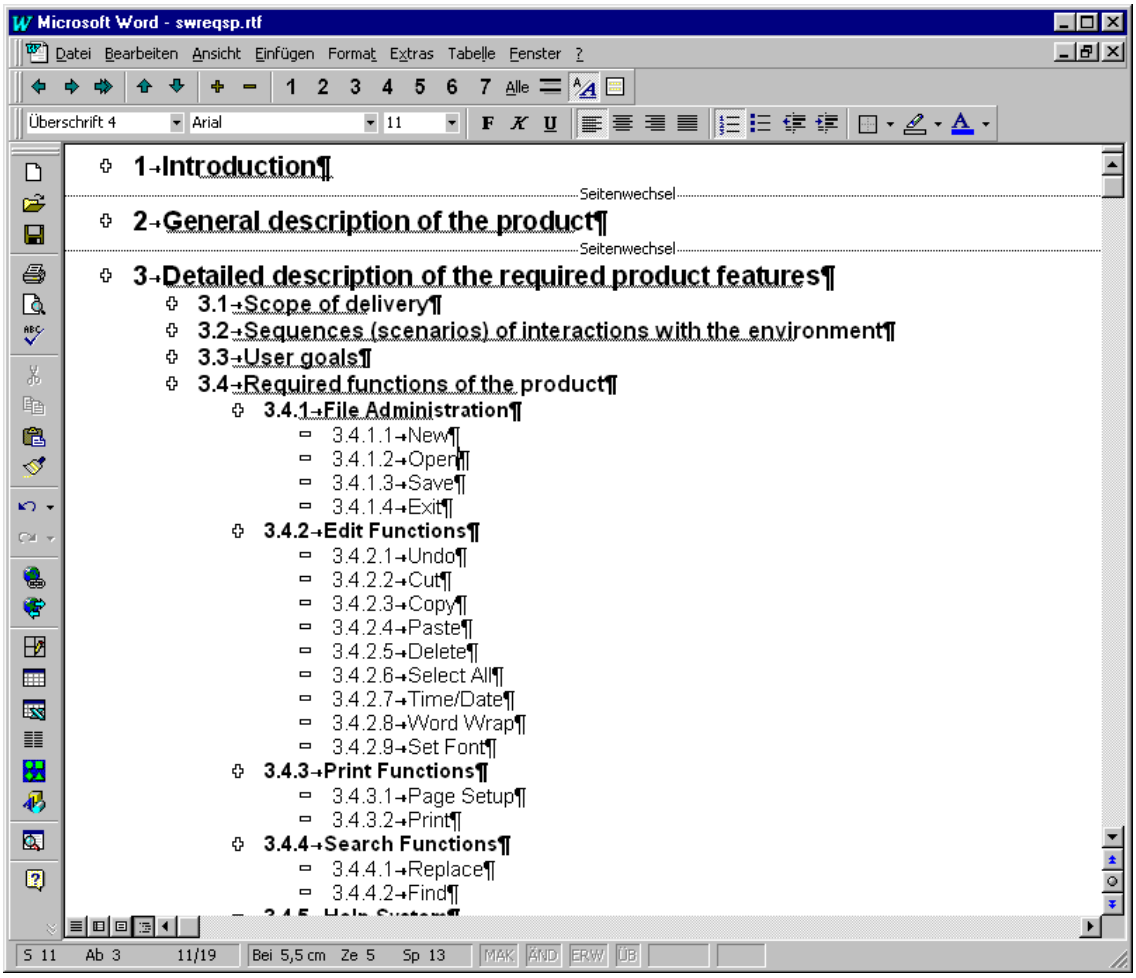
Figure 11 - Reopen a test structure

3.1.2.2.4Importing an RTF-formatted document

TEMPPO is able to generate test packages and test cases from headings within RTF-formatted documents. You can either generate a basic test structure containing just test packages and no test cases or even a complete test structure with test cases.

3.1.2.2.4.1 Generating a basic test structure

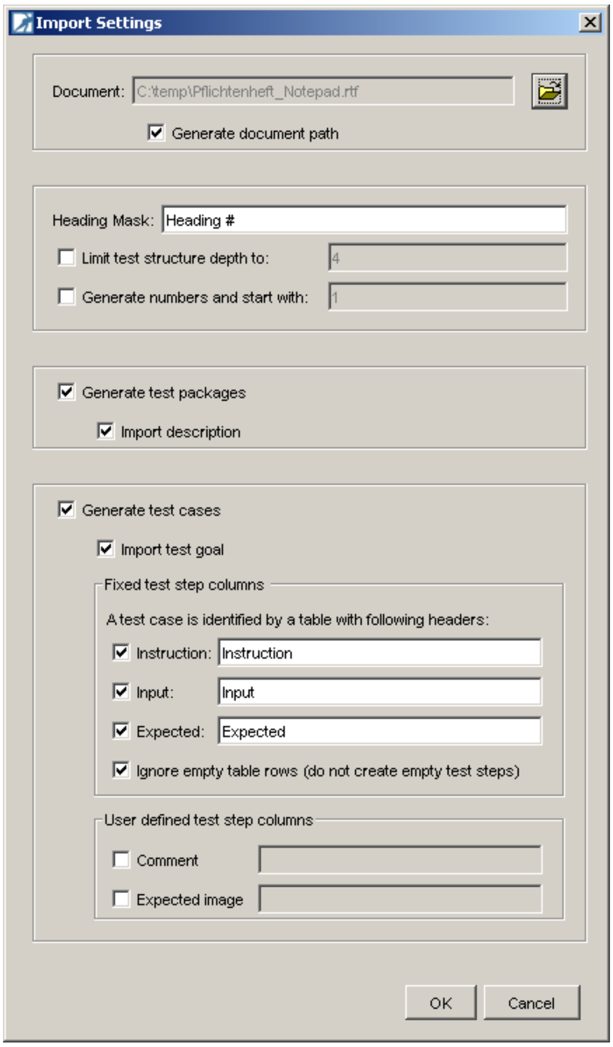
If requirements are specified in Software Requirement Specification (SRS) documents, they can be used to generate a basic test structure. The obvious advantage of creating a test structure according to the structure of a base document is that the test case-to-requirement relation is evident every time. This enables the test responsible to easily verify, which requirements are covered by test cases.


Microsoft Word - swreqsp.rtf
Datei Bearbeiten Ansicht Einfügen Format Extras Tabelle Fenster ?
1 2 3 4 5 6 7 Alle
Überschrift 4
Arial
11
F K U
Seitenwechsel
Seitenwechsel
S 11 Ab 3 11/19 Bei 5,5 cm Ze 5 Sp 13 MAK ÄND ERW ÜB

- 1→Introduction¶
- 2→General description of the product¶
- 3→Detailed description of the required product features¶
  - 3.1→Scope of delivery¶
  - 3.2→Sequences (scenarios) of interactions with the environment¶
  - 3.3→User goals¶
  - 3.4→Required functions of the product¶
    - 3.4.1→File Administration¶
      - 3.4.1.1→New¶
      - 3.4.1.2→Open¶
      - 3.4.1.3→Save¶
      - 3.4.1.4→Exit¶
    - 3.4.2→Edit Functions¶
      - 3.4.2.1→Undo¶
      - 3.4.2.2→Cut¶
      - 3.4.2.3→Copy¶
      - 3.4.2.4→Paste¶
      - 3.4.2.5→Delete¶
      - 3.4.2.6→Select All¶
      - 3.4.2.7→Time/Date¶
      - 3.4.2.8→Word Wrap¶
      - 3.4.2.9→Set Font¶
    - 3.4.3→Print Functions¶
      - 3.4.3.1→Page Setup¶
      - 3.4.3.2→Print¶
    - 3.4.4→Search Functions¶
      - 3.4.4.1→Replace¶
      - 3.4.4.2→Find¶
    - 3.4.5→Help System¶

Figure 12 – Sample Software Requirements Specification for MS Notepad

Before starting the import process, an empty test structure (see 3.1.2.2.1) has to be created. Then the menu item **Test Structure > Import Document Structure** has to be activated, which will cause the `Import Settings` dialog to be displayed.



Document: C:\temp\Pflichtenheft_Notepad.rtf

☑ Generate document path

Heading Mask: Heading #

☐ Limit test structure depth to: 4

☐ Generate numbers and start with: 1

☑ Generate test packages

☑ Import description

☑ Generate test cases

☑ Import test goal

Fixed test step columns

A test case is identified by a table with following headers:

☑ Instruction: Instruction

☑ Input: Input

☑ Expected: Expected

☑ Ignore empty table rows (do not create empty test steps)

User defined test step columns

☐ Comment

☐ Expected image

Figure 13 - Import Settings for generating a basic test structure

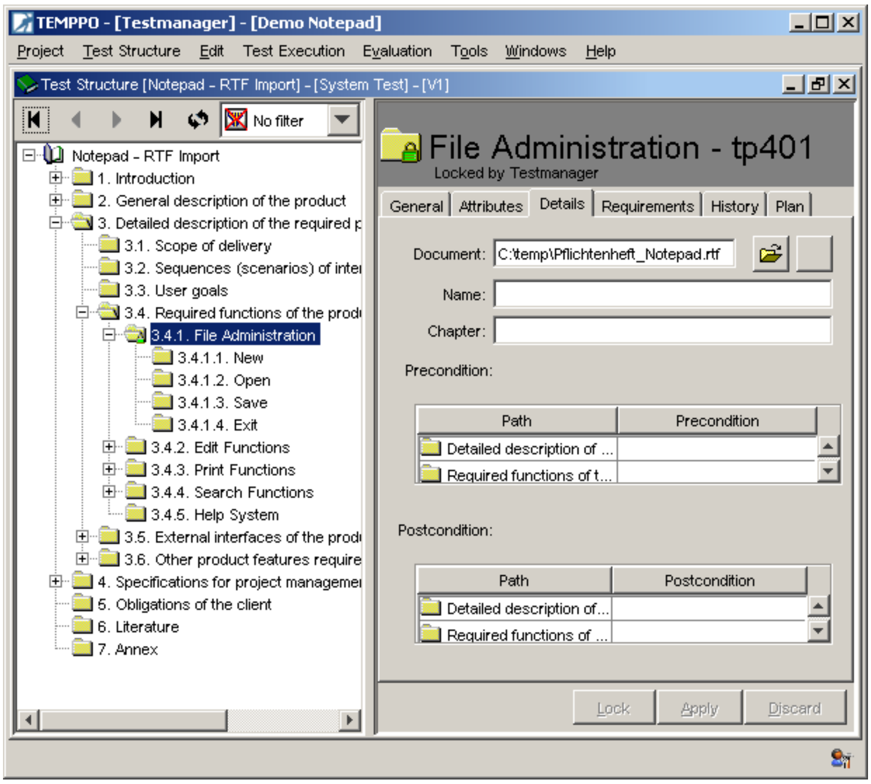
For the generation of a basic test structure following settings can be applied:

- **Generate document path**: The path of the imported document is set for each test package in tab **Details**.
- **Heading mask**: Specify the name of the heading (if it is different to the standard one: [Heading | Überschrift] 1,2,3,…). It is initialized with "*Heading #*".
- **Limit test structure depth to**: If checked, limits the imported document hierarchy to the specified level. This means, that headings with a level higher than the specified will not be imported.
- **Generate numbers and start with**: If checked, a hierarchical numbering will be generated for each test package and added as a prefix to its name (like heading numbering in MS Word).
- **Generate test packages**: If checked, test packages will be generated. Test packages will be generated from document paragraphs, which are formatted with the standard formats "Heading 1" – "Heading 9" (or "Überschrift 1" – "Überschrift 9" in German documents). Their hierarchical position in the test structure will correspond to the heading level in the document.
- **Import description**: enabled, if checkbox '"**Generate Test Packages"** is checked. If checked, the test package description will be imported. Tables and graphics are not considered.

All other settings are not relevant when generating a basic test structure and will be explained later.

Importing the document displayed in Figure 12 with settings displayed in Figure 13 will generate the test structure displayed in Figure 14.

Please consider that importing is applied on the selected test package!


TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Structure [Notepad - RTF Import] - [System Test] - [V1]
No filter
Notepad - RTF Import
1. Introduction
2. General description of the product
3. Detailed description of the required p
3.1. Scope of delivery
3.2. Sequences (scenarios) of inter
3.3. User goals
3.4. Required functions of the produ
3.4.1. File Administration
3.4.1.1. New
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.5. Help System
3.5. External interfaces of the produ
3.6. Other product features require
4. Specifications for project manageme
5. Obligations of the client
6. Literature
7. Annex
File Administration - tp401
Locked by Testmanager
General
Attributes
Details
Requirements
History
Plan
Lock
Apply
Discard

Document: C:\temp\Pflichtenheft_Notepad.rtf

Name:

Chapter:

Precondition:

Path	Precondition
Detailed description of ...	
Required functions of t...	

Postcondition:

Path	Postcondition
Detailed description of...	
Required functions of ...	

Figure 14 - Generated basic test structure

Headings above level 3 (e.g. 3.4.1.1 – 3.4.1.4) were not imported, due to limiting the test structure depth to 3. A numbering was added to the test package names, because of checking *generate numbers and start with.* A reference to the imported document was applied to all generated test packages, which made them be displayed with the yellow folder icon.

TEMPPO generates test packages for the whole document structure. Of course you'll never create any test case for chapters like Introduction, Annex ... so you should delete the corresponding test packages.

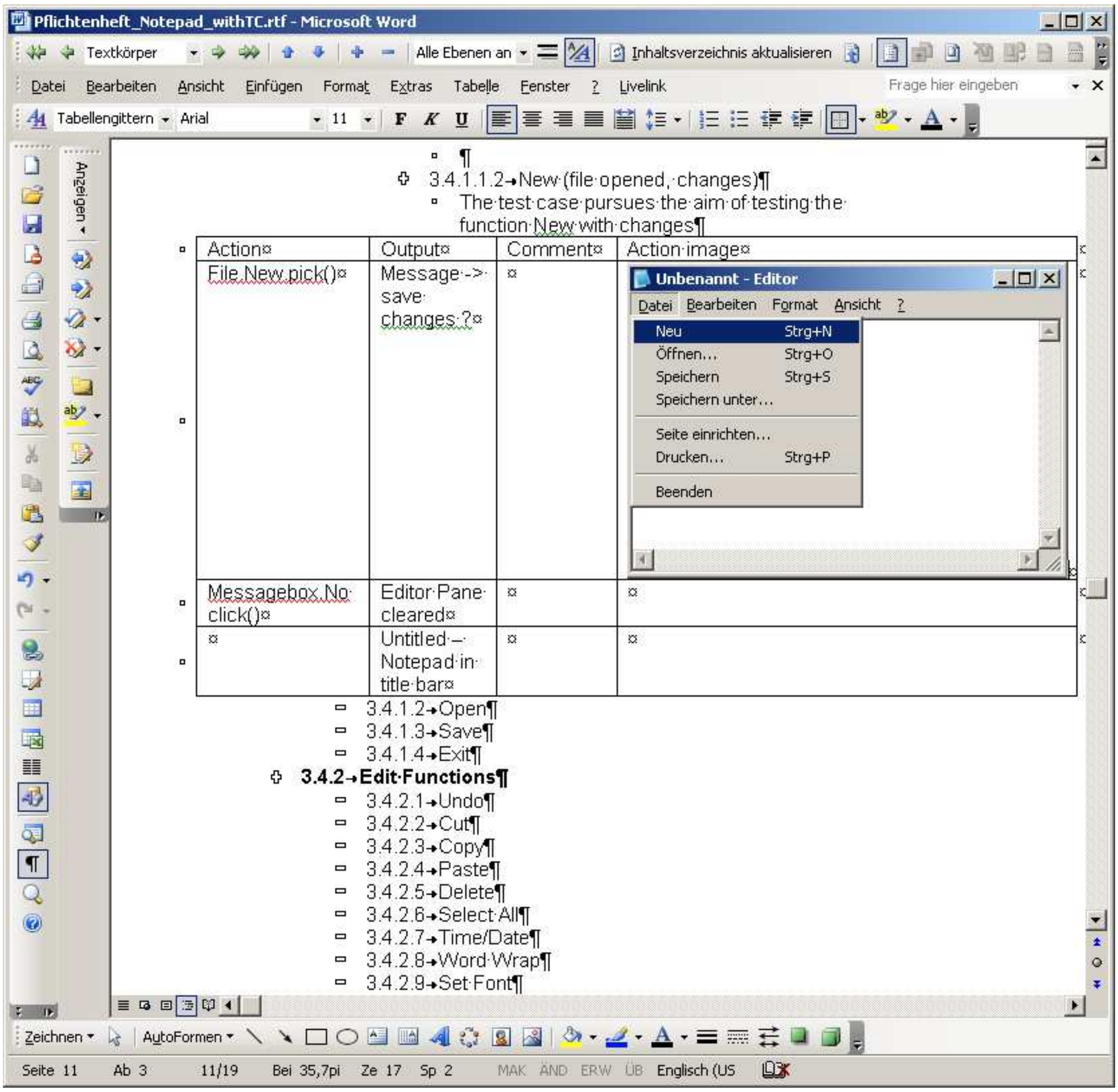
Such an initial test structure may be extended by adding more test packages, if necessary. For each package that is a leave in the test structure tree, at least one test case should be defined.

TIP for an import: If an rtf-document gets very huge (e.g. embedded graphical objects: 50MB), TEMPPO is not able to read in such a huge WORD document. Replace all the graphical objects with "nothing" and the import of this rtf document will work!

3.1.2.2.4.2 Generating a complete test structure

In certain circumstances you may have already written down your test cases in a document (e.g. test case specification). This happens if you don't have a "test management tool" other than a text editor. When the time has come to introduce TEMPPO to your project, obviously you don't want to create the test structure and test cases in your document a second time. For this purpose TEMPPO offers an additional importing feature that enables the generation of test cases.

ⓘ **When generating test cases from RTF documents, you mix the phases design and implementation. It's very important to consider this fact, because you still have to carry out tasks from the design phase after test case import!**



Pflichtenheft_Notepad_withTC.rtf - Microsoft Word
Textkörper
Alle Ebenen an
Inhaltsverzeichnis aktualisieren
Datei Bearbeiten Ansicht Einfügen Format Extras Tabelle Fenster ? Livelink
Frage hier eingeben
Tabellengittern
Arial
11
Anzeigen
Unbenannt - Editor
Datei Bearbeiten Format Ansicht ?
Neu Strg+N
Öffnen... Strg+O
Speichern Strg+S
Speichern unter...
Seite einrichten...
Drucken... Strg+P
Beenden
Zeichnen
AutoFormen
Seite 11 Ab 3 11/19 Bei 35,7pi Ze 17 Sp 2 MAK ÄND ERW ÜB Englisch (US

3.4.1.1.2 New (file opened, changes)

The test case pursues the aim of testing the function New with changes

Action	Output	Comment	Action image
File.New.pick()	Message -> save changes?		
Messagebox.No click()	Editor Pane cleared		
	Untitled – Notepad in title bar		

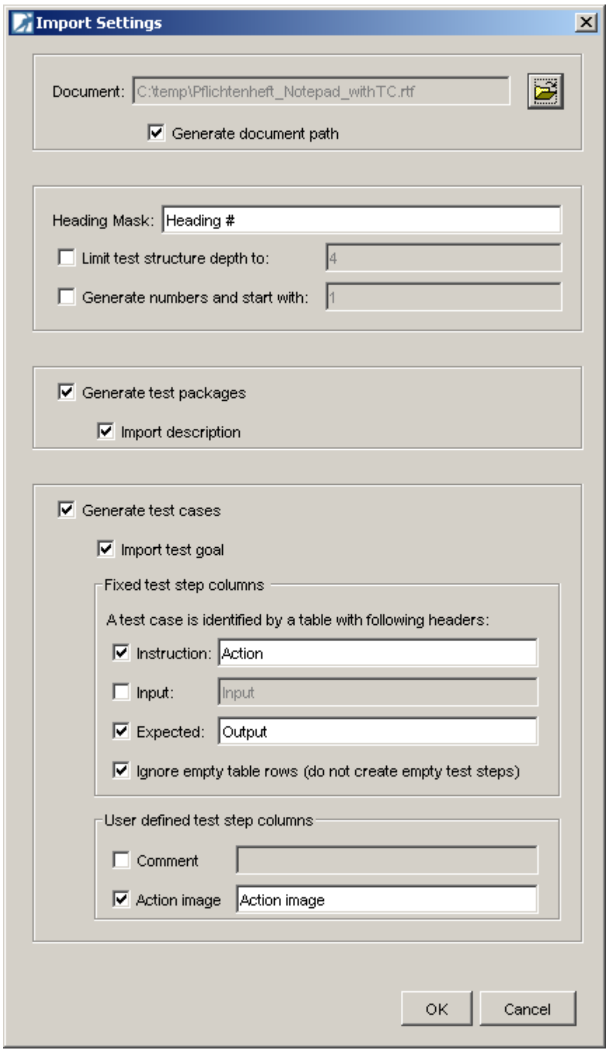
3.4.1.2 Open
3.4.1.3 Save
3.4.1.4 Exit

3.4.2 Edit Functions

3.4.2.1 Undo
3.4.2.2 Cut
3.4.2.3 Copy
3.4.2.4 Paste
3.4.2.5 Delete
3.4.2.6 Select All
3.4.2.7 Time/Date
3.4.2.8 Word Wrap
3.4.2.9 Set Font

Figure 15 – Test Case Specification

Like described in 3.1.2.2.4.1 open the **Import Settings** by activating **Test Structure > Import Document Structure**.



Import Settings

Document: C:\temp\Pflichtenheft_Notepad_withTC.rtf

☑ Generate document path

Heading Mask: Heading #

□ Limit test structure depth to: 4

□ Generate numbers and start with: 1

☑ Generate test packages

☑ Import description

☑ Generate test cases

☑ Import test goal

Fixed test step columns

A test case is identified by a table with following headers:

☑ Instruction: Action

□ Input: Input

☑ Expected: Output

☑ Ignore empty table rows (do not create empty test steps)

User defined test step columns

□ Comment

☑ Action image: Action image

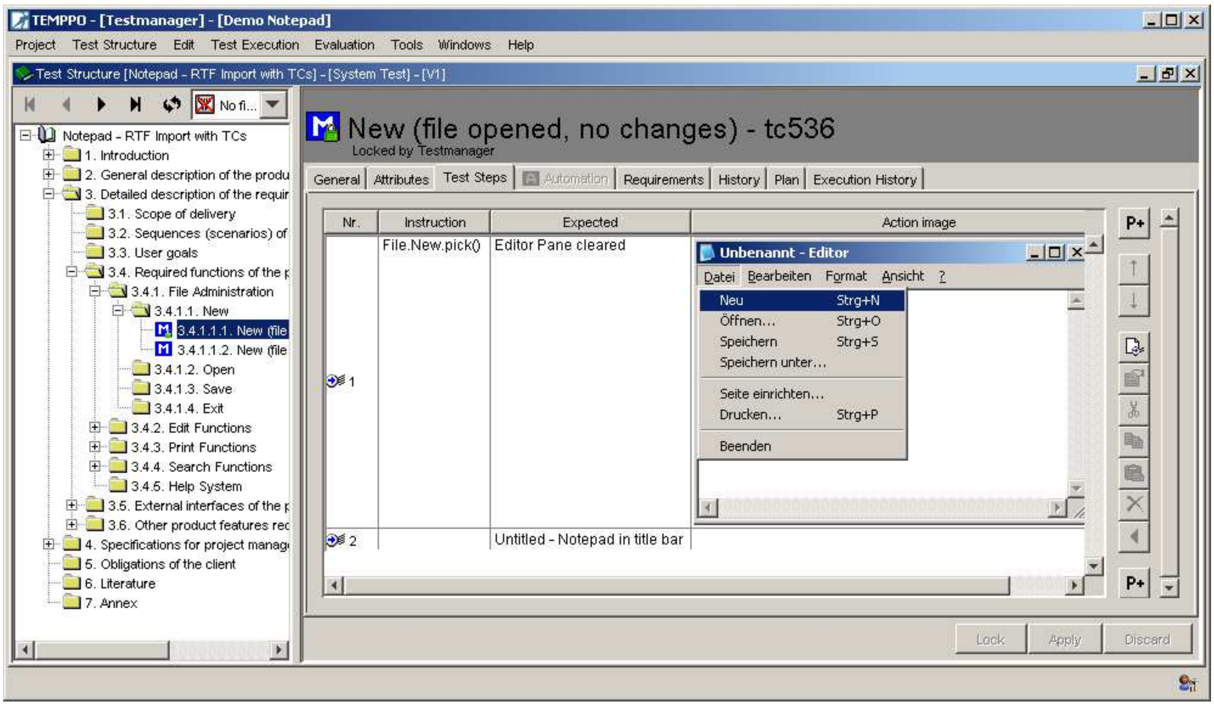
Figure 16 - Import Settings for generating a complete test structure

Additionally to the settings illustrated in Figure 13 you can do the following:

- **Generate test cases**: If checked, test cases will be generated. Test cases will be generated from document paragraphs, which are formatted with the standard formats "Heading 1" – "Heading 9" (or "Überschrift 1" – "Überschrift 9" in German documents). Additionally the paragraph after the heading which should become a test case must contain a test step table. This table must contain columns which describe **instruction**, **input** and **expected** output of test steps. These columns may have arbitrary headers, like "Action" for the instruction (see Figure 16). Furthermore you can suppress the generation of empty test steps in case of empty table rows by checking "**ignore empty table rows**".
  The hierarchical position of the test case in the test structure will correspond to the heading level in the document.
- **Import test goal**: Enabled, if checkbox "**Generate test cases**" is checked. If checked, all the text after the heading until the next heading will be imported to the test case's **Test goal**.
- **User defined test step columns:** Since TEMPPO offers not only the 3 standard columns, user can define additional columns that are also used for import.

Importing the document displayed in Figure 15 with settings displayed in Figure 16 will generate the test structure displayed in Figure 17.

Please consider that importing is applied on the selected test package!



Nr.	Instruction	Expected	Action image
1	File.New.pick()	Editor Pane cleared	
2		Untitled - Notepad in title bar	

Figure 17 – Generated complete test structure

Test cases were generated due to the tables in the chapters 2.1.1.1 and 2.1.1.2. The data in column **Action** was used for test step **Instruction**, the data in column **Output** for **Expected**, the image for column **Action Image**. The column **Comments** was ignored. A reference to the imported document was applied to all generated test packages, which made them be displayed with the yellow folder icon.

This test structure can still be extended by adding more test packages or test cases, if necessary. For each package that is a leaf in the test structure tree, at least one test case should be defined.

ⓘ **Of course, TEMPPO is not able to verify the correctness of your document. So please ensure, that you’ve done the correct settings and your document fulfills the needs for importing, such as headings formatted with “Heading 1” – “Heading 9” (or “Überschrift 1” – “Überschrift 9” in German documents) and proper test step tables.**

3.1.2.2.5Import Test Cases from Excel

TEMPPO is able to import test cases from an Excel list. First you have to save your Excel file as csv and select a test package.

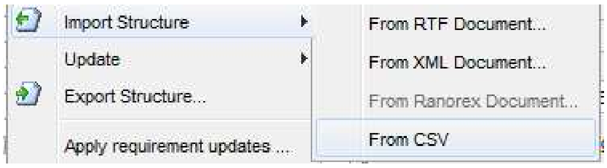


Figure 18 - Menu - Import Test Cases from Excel

When activating the menu Test Structure -> Import Structure -> From CSV Figure 19 is displayed. In that dialog you can assign Excel to TEMPPO attributes.

Mandatory attributes: Name and ID have to be assigned, otherwise an import cannot be performed.

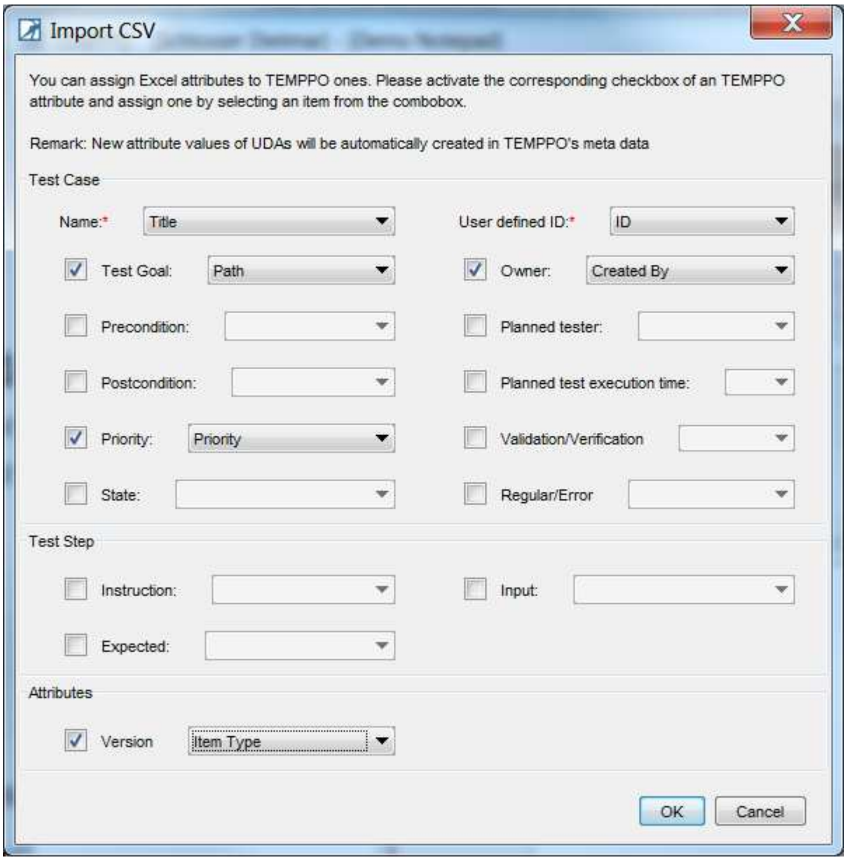
Fixed attributes like test goal, state…: can be assigned to any Excel attributes. Attribute values have to be identically. If they're different, the default value will be set.

If the **owner** is selected and the user is not available in TEMPPO, the current importer is used as owner.

Test step: The 3 attributes Instruction, Expected, Input can be assigned with exactly 1 step.

User Defined Attributes: Any UDAs can be assigned with Excel attributes. If a value to be assigned doesn't exist, it will be created automatically.

To assign the attributes above, please click the checkbox.



Import CSV

You can assign Excel attributes to TEMPPO ones. Please activate the corresponding checkbox of an TEMPPO attribute and assign one by selecting an item from the combobox.

Remark: New attribute values of UDAs will be automatically created in TEMPPO's meta data

Test Case

Name:* Title
User defined ID:* ID
☑ Test Goal: Path
☑ Owner: Created By
□ Precondition:
□ Planned tester:
□ Postcondition:
□ Planned test execution time:
☑ Priority: Priority
□ Validation/Verification
□ State:
□ Regular/Error

Test Step

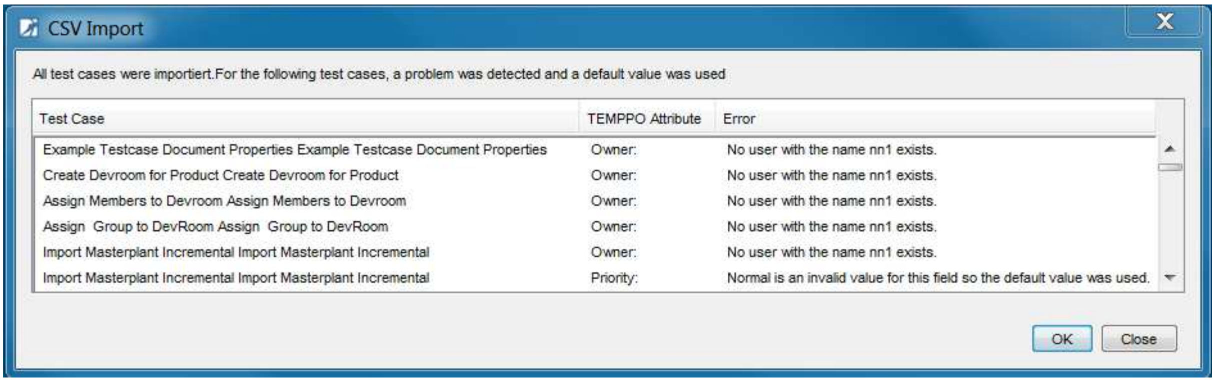
□ Instruction:
□ Input:
□ Expected:

Attributes

☑ Version Item Type

Figure 19 - Assign Excel to TEMPPO attributes

After pressing OK the user is informed about errors of the import.



All test cases were importiert.For the following test cases, a problem was detected and a default value was used

Test Case	TEMPPO Attribute	Error
Example Testcase Document Properties Example Testcase Document Properties	Owner:	No user with the name nn1 exists.
Create Devroom for Product Create Devroom for Product	Owner:	No user with the name nn1 exists.
Assign Members to Devroom Assign Members to Devroom	Owner:	No user with the name nn1 exists.
Assign Group to DevRoom Assign Group to DevRoom	Owner:	No user with the name nn1 exists.
Import Masterplant Incremental Import Masterplant Incremental	Owner:	No user with the name nn1 exists.
Import Masterplant Incremental Import Masterplant Incremental	Priority:	Normal is an invalid value for this field so the default value was used.

Figure 20 - CSV Import Result

3.1.2.2.6Build manually

If the requirement specification is not imported to TEMPPO, the user has to create test packages manually (activate the menu item **Edit > New > Test Package**). It is strongly recommended for each test package to specify a document chapter, where the respective requirement is defined. Therefore in the next chapters only a test structure based on a requirement specification is considered.

3.1.2.2.7XML Import

TEMPPO is able to generate a whole test structure with test packages, test cases, metadata (requirements, attributes) and all the information of test cases etc. by importing an XML file.

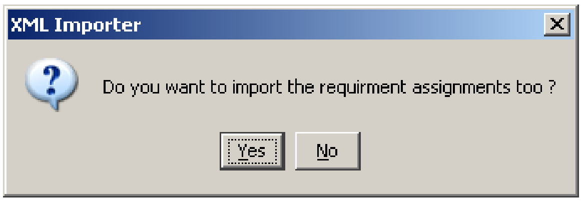
The XML-Import can be started by pressing the menu item **Test Structure > Import Structure > From XML Document…**. This menu item is only enabled when a test structure is open and selected or a test package is selected.

The XML-document to be imported can then be chosen in a file dialog. The XML document is checked against the DTD (Document Type Definition) (See 13.1). Your XML file has to have the first line

<!DOCTYPE TEMPPO_EXCHANGE SYSTEM "Test-structure.dtd" > whereas Test-structure.dtd is the DTD file (See 13.1). Otherwise the import will fail.

ⓘ **In your TEMPPO installation directory there is a subdirectory XML, where you'll find the DTD (Test-structure.dtd) together with an XML example.**

Before the importing process can start, you have to decide if the requirement assignments should be imported, too.



Do you want to import the requirment assignments too ?

Figure 21 – Import with requirement assignments

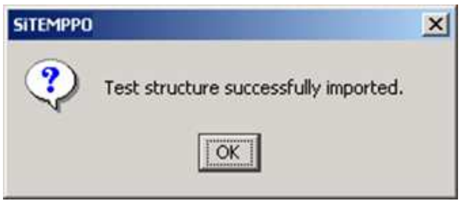
ⓘ **Before you can import a test structure with requirement assignments correctly, the requirement structure already has to exist in the database. Import the requirement structure before.**

Stages of Import:

1. Parsing and validation of XML-file, checking of semantics and creation of new objects
2. Saving new or modified objects to the database
3. Reloading of the whole test structure from database

Each stage has its own progress bar.

When the import is finished, the following information dialog comes up:



Test structure successfully imported.

The imported test packages and cases are placed below the selected node.

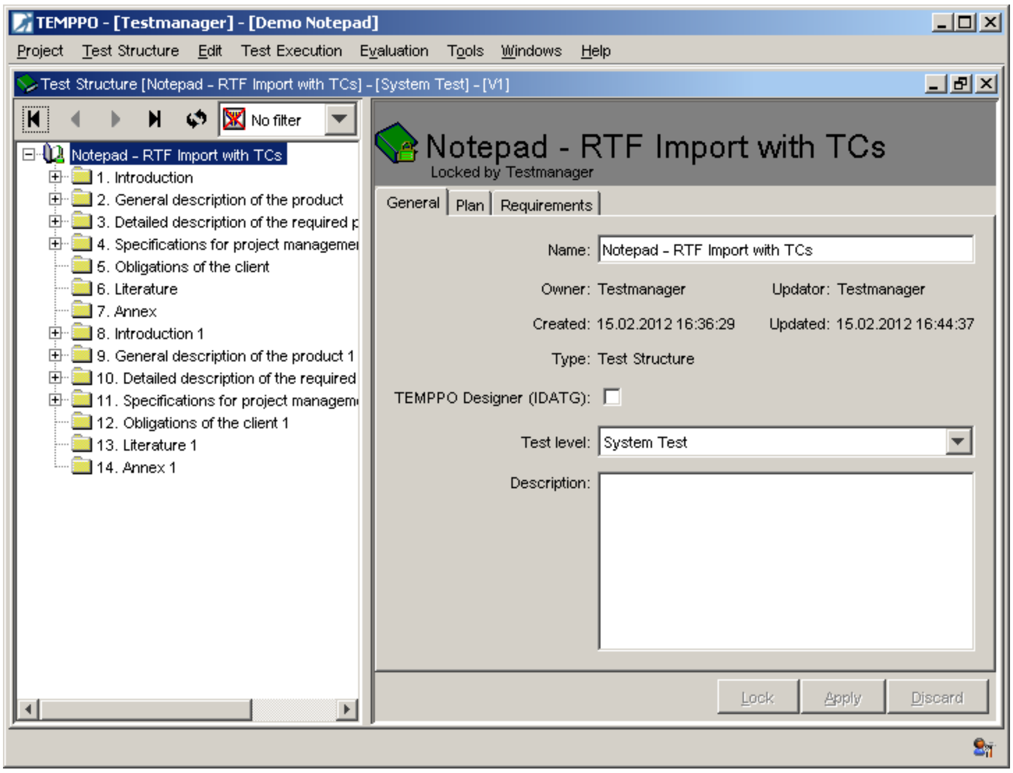
3.1.2.2.7.1 Possible errors during the import:

Errors	Reaction
XML file doesn't match DTD schema.	Nothing is written to the database, a message is shown and the project status before the import is loaded from database.
Required attribute exists but is zero-length.	a) Test case name, Test package name: a name is generated automatically ("New Test Package", "New Test Case"). No Message is shown. b) User defined attributes: Message is shown. The user can continue the import (invalid element ignored) or stop import and return to the previous state.
The requirement structure can't be found in the database.	A message is shown. The name of the requirement structure which released an error is written in the error file. If the requirement structure doesn't exist in the database, you have to import it before importing the test structure. If the name of the requirement structure isn't defined correctly in the input file, you have to correct it. Now try it again.
A requirement can't be found in database.	A message is shown (see Figure 22). The user defined id of the requirement which released an error is written in the error file. Please correct the name in the input file and then try it again.

Figure 22 – Information message – error with requirements

3.1.2.2.7.2 Special cases

Special cases	**Reaction**
Name of test package, test case or user-defined ID exists.	Unique name is generated like “ 1”, “ 2”. No message is shown. (See Figure 23)
Test level, category, attribute name or values doesn’t exist.	Object is created. No message is shown.
Attribute name exists, new value in XML file	Value is added to attribute



Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required p
4. Specifications for project manageme
5. Obligations of the client
6. Literature
7. Annex
8. Introduction 1
9. General description of the product 1
10. Detailed description of the required
11. Specifications for project managem
12. Obligations of the client 1
13. Literature 1
14. Annex 1

Notepad - RTF Import with TCs
Locked by Testmanager

Name: Notepad - RTF Import with TCs
Owner: Testmanager Updator: Testmanager
Created: 15.02.2012 16:36:29 Updated: 15.02.2012 16:44:37
Type: Test Structure
TEMPPO Designer (IDATG): □
Test level: System Test
Description:

Figure 23 – name of test package already exists

3.1.2.2.7.3 Transfer data via XML

Import the test structure from one project P1 to another project p2, same database

- Export the test structure via XML (TEMPPO Test Manager)
- Assign the needed requirement structures of the test structures to the project P2 (TEMPPO Administrator)
- Import the test structure via XML (TEMPPO Test Manager)

Import the test structure from one database d1(export) to another database d2 (project p2, import)

- Export the test structure via XML (TEMPPO Test Manager)
- Export the requirement structures in the needed (!!) version (TEMPPO Requirement Manager)

- Import the requirement structures via XML into database d2 (TEMPPO Requirement Manager)
- Check In the requirement structures (TEMPPO Requirement Manager)
- Assign the needed requirement structures of the test structures to the project P2 (TEMPPO Administrator)
- Import the test structure via XML (TEMPPO Test Manager)

3.1.2.3 Extending the Test Structure

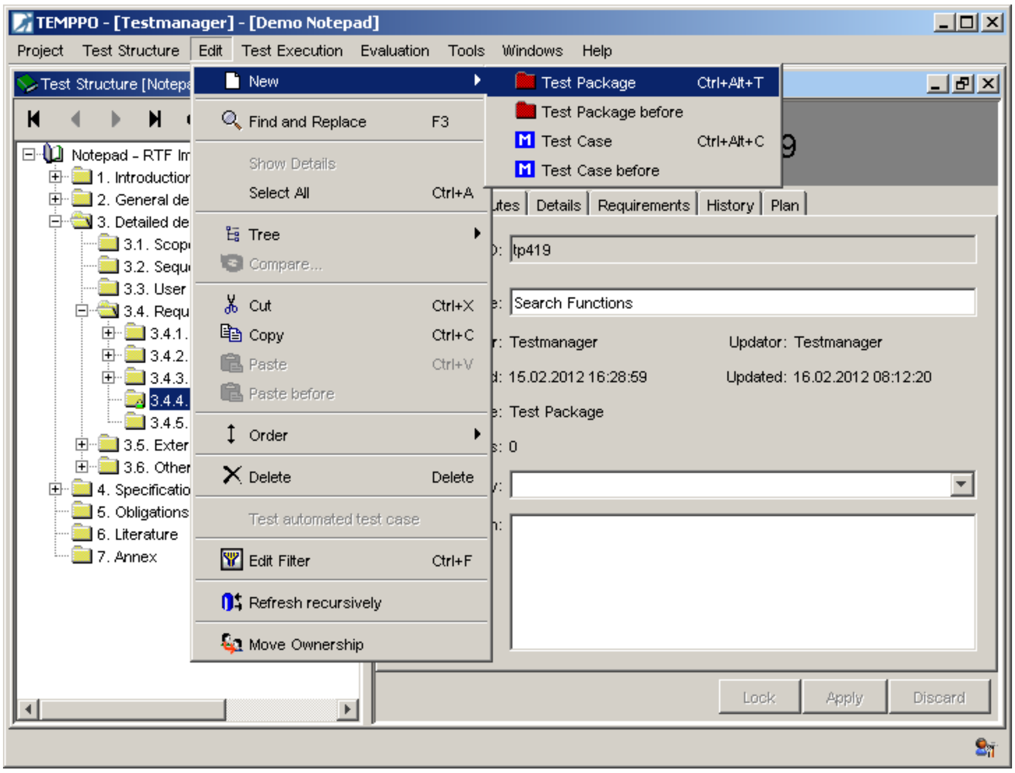
Sometimes quite a few chapters in the requirement specification contain a lot of information. During test structure generation you will of course get just one test package for each such chapter. The problem is, that you will have to create a lot of test cases for these test packages (due to the abundant information), which will lead to a badly arranged test structure. The solution is to split them up. Such a refinement can be done by adding further test packages.

3.1.2.3.1Adding Test Packages

New sub-test packages can be added by activating the menu item **Edit > New > Test Package** or **Edit > New > Test Package before**. A new child node is created below or before the selected one and a new name has to be entered. The default name is **New Test Package**.

Let’s consider the following example:

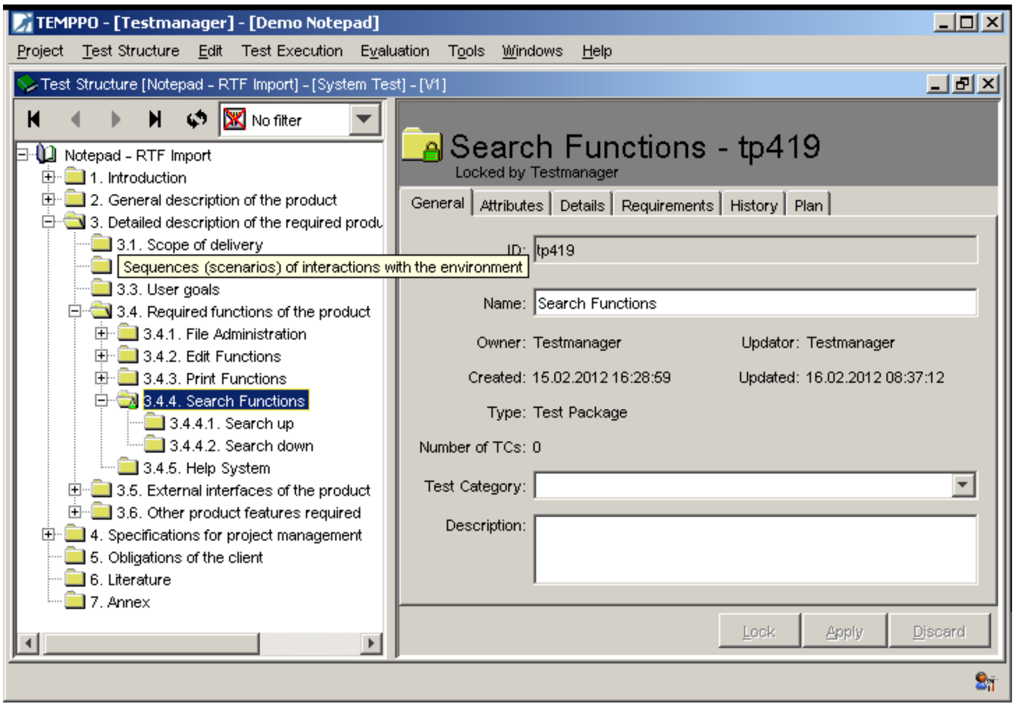
> The test structure is created manually. The requirement **Search Functions** consists of 2 sub-requirements, each described in exactly one chapter. Therefore, 2 sub-test packages are necessary that refer to these requirements.



): tp419
e: Search Functions
r: Testmanager
Updator: Testmanager
d: 15.02.2012 16:28:59
Updated: 16.02.2012 08:12:20
e: Test Package
s: 0

Figure 24 - Create new test package

In Figure 25 such a test package **Search Functions** was identified and split up into **Search Up** and **Search Down**. A reference to the document containing the description (e.g. a use case document) of the respective requirement was applied to the new test packages, which made them be displayed colored yellow.



Notepad - RTF Import
1. Introduction
2. General description of the product
3. Detailed description of the required produ
3.1. Scope of delivery
Sequences (scenarios) of interactions with the environment
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Search up
3.4.4.2. Search down
3.4.5. Help System
3.5. External interfaces of the product
3.6. Other product features required
4. Specifications for project management
5. Obligations of the client
6. Literature
7. Annex

Search Functions - tp419
Locked by Testmanager

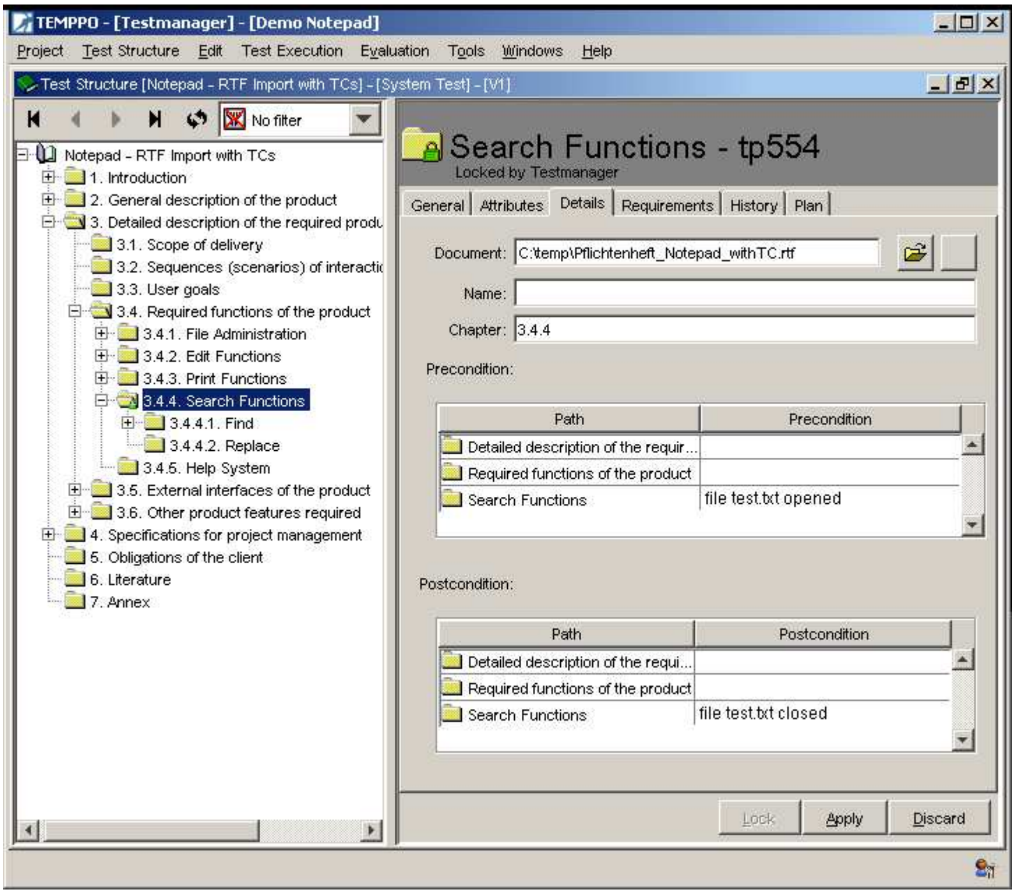
ID: tp419
Name: Search Functions
Owner: Testmanager
Updator: Testmanager
Created: 15.02.2012 16:28:59
Updated: 16.02.2012 08:37:12
Type: Test Package
Number of TCs: 0
Test Category:
Description:

Figure 25 - Adding new test packages that are directly related to requirements

Now consider the opposite situation:

> The functionalities for searching up or down are not described in separate chapters of the requirements specification.

In this case, you also have the possibility to "split" the requirement (e.g. "Search Functions") into two test packages. This is the suggested way to create a "good" test structure. Test packages will keep their red color, since there is no chapter that is directly related to them.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required produ
3.1. Scope of delivery
3.2. Sequences (scenarios) of interacti
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of the product
3.6. Other product features required
4. Specifications for project management
5. Obligations of the client
6. Literature
7. Annex
Search Functions - tp554
Locked by Testmanager
General Attributes Details Requirements History Plan
Lock Apply Discard

Document: C:\temp\Pflichtenheft_Notepad_withTC.rtf

Name:

Chapter: 3.4.4

Precondition:

Path	Precondition
Detailed description of the requir...	
Required functions of the product	
Search Functions	file test.txt opened

Postcondition:

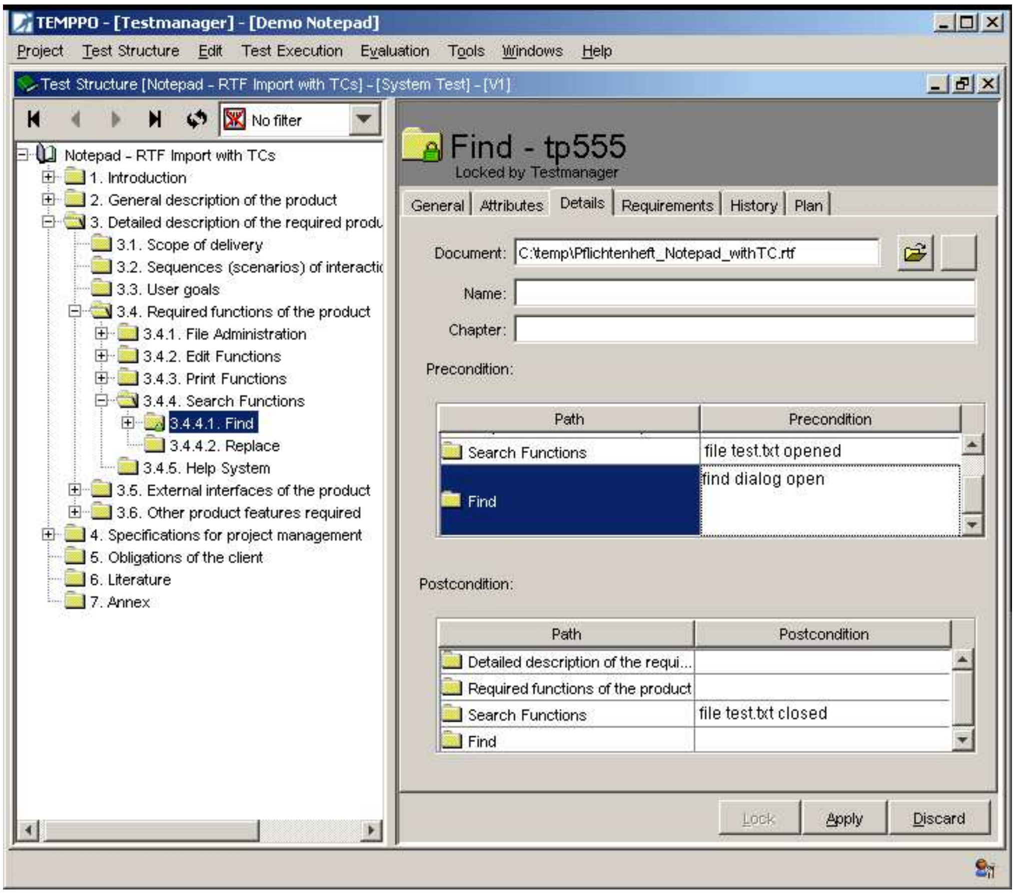
Path	Postcondition
Detailed description of the requi...	
Required functions of the product	
Search Functions	file test.txt closed

Figure 26 –Adding new test packages without direct relations to document chapter

ⓘ **You can add test packages that do or don't refer to requirements in random hierarchical order. It is up to you, if the structure remains comprehensible.**

When activating the tab **Details**, a special **precondition** and a special **postcondition** for this test package may be specified. This is suggested, if all contained test cases have a common “setup” phase.

In the table below all preconditions and postconditions of the parent test packages are listed.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required produ
3.1. Scope of delivery
3.2. Sequences (scenarios) of interacti
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of the product
3.6. Other product features required
4. Specifications for project management
5. Obligations of the client
6. Literature
7. Annex
Find - tp555
Locked by Testmanager
General Attributes Details Requirements History Plan
Lock Apply Discard

Document: C:\temp\Pflichtenheft_Notepad_withTC.rtf

Name:

Chapter:

Precondition:

Path	Precondition
Search Functions	file test.txt opened
Find	find dialog open

Postcondition:

Path	Postcondition
Detailed description of the requi...	
Required functions of the product	
Search Functions	file test.txt closed
Find	

Figure 27 – Precondition and Postcondition for test package

3.1.2.3.2Categorizing Test Packages

After creating a test package you can select a **test category**. Here all categories are available that are connected to the current test level (for further details of test categories see 5.4.3.). These are some typical test categories:

- Task-Based Test
- GUI-Layout Test
- State-Based Test

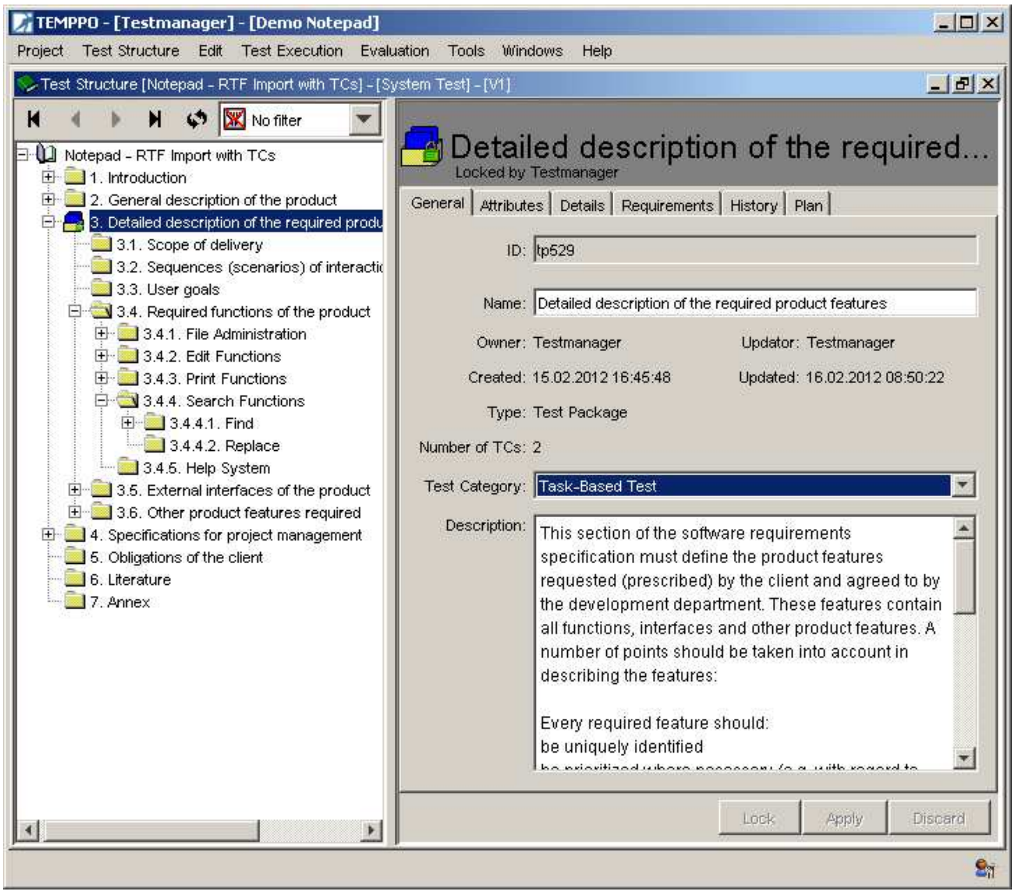
In Figure 28 the test category **Task-Based Test**” was assigned to a test package. After assigning a test category the node looks like .

At first sight assigning test categories seems to be superfluous, because they have no further influence on the test structure than being a flag for a test package. A closer inspection makes their purpose plain. Firstly, you can obtain statistical information from your test structure or test suite, like:

- “How many functional or performance test cases does my test structure contain?”
- “How many functional test cases are failed?”

For further details on analyses see 3.1.5.1.

Secondly, you are able to create test suites based on test categories, e.g. one for all functional test cases (see 3.1.4.1).



Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required produ
3.1. Scope of delivery
3.2. Sequences (scenarios) of interacti
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of the product
3.6. Other product features required
4. Specifications for project management
5. Obligations of the client
6. Literature
7. Annex

Detailed description of the required...
Locked by Testmanager

ID: tp529

Name: Detailed description of the required product features

Owner: Testmanager
Updator: Testmanager

Created: 15.02.2012 16:45:48
Updated: 16.02.2012 08:50:22

Type: Test Package

Number of TCs: 2

Test Category: Task-Based Test

Description: This section of the software requirements specification must define the product features requested (prescribed) by the client and agreed to by the development department. These features contain all functions, interfaces and other product features. A number of points should be taken into account in describing the features:

Every required feature should:
be uniquely identified

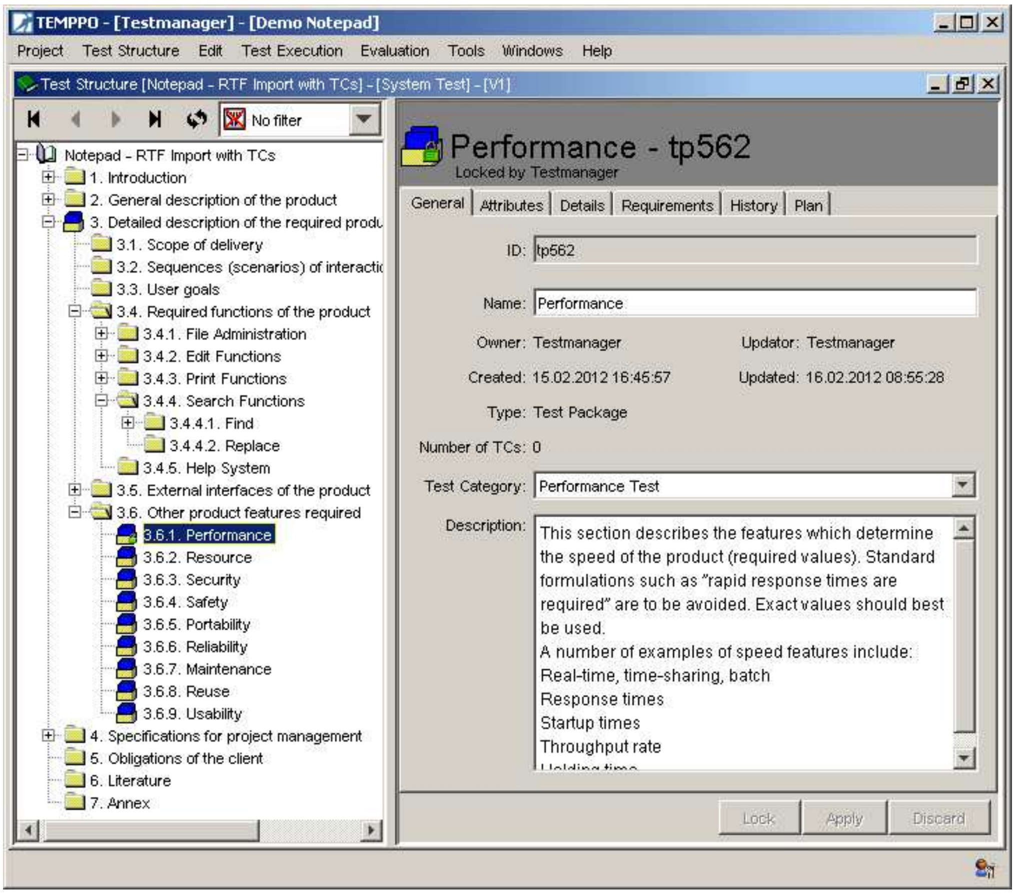
Figure 28 - Assigning a test category

When creating test packages, you may determine appropriate test categories. Normally you start test case design with the chapter **Required functions of the product** that should cover all task-based tests.

In further steps of test case design, you identify GUI-layout tests in chapter “External interfaces – User interface”. Such test packages should be given the test category **GUI-layout test**. They pursue the aim to cover tests for input syntax, check completeness of GUI-masks etc.

Furthermore you categorize the test packages below “Other product features required” that cover mandatory quality attributes like performance, mass or stress test.

The result of the steps above is displayed in Figure 29.



ID: tp562

Name: Performance

Owner: Testmanager   Updator: Testmanager

Created: 15.02.2012 16:45:57   Updated: 16.02.2012 08:55:28

Type: Test Package

Number of TCs: 0

Test Category: Performance Test

Description: This section describes the features which determine the speed of the product (required values). Standard formulations such as "rapid response times are required" are to be avoided. Exact values should best be used.
A number of examples of speed features include:
Real-time, time-sharing, batch
Response times
Startup times
Throughput rate
Holding time

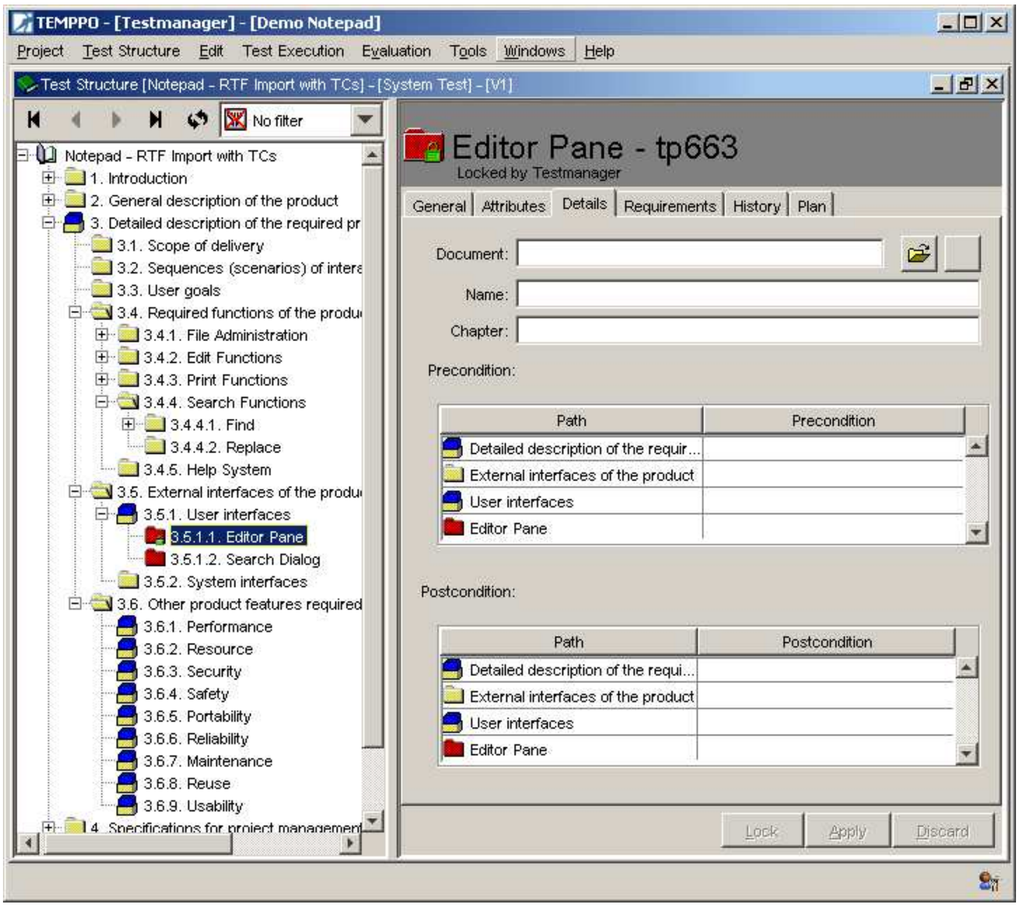
Figure 29 – Complete Test Structure with Test Categories

3.1.2.3.3 Referring to documents

Test packages can refer to a chapter in some document (e.g. Software Requirement Specification), which usually defines some requirement(s).

A test package is initially colored red, but as soon as it refers to a document, its color changes to yellow. This happens, if at least one of the following attributes is specified:

- document file
- document name
- chapter



Path	Precondition
Detailed description of the requir...	
External interfaces of the product	
User interfaces	
Editor Pane	

Path	Postcondition
Detailed description of the requi...	
External interfaces of the product	
User interfaces	
Editor Pane	

Figure 30 - Test package without reference to a document, name or chapter

First of all you create a new test structure or open an existing one. After this the test structure can be filled in the following ways:

– Importing an RTF-formatted document
– Import Test Cases from Excel

TEMPPO is able to import test cases from an Excel list. First you have to save your Excel file as csv and select a test package.

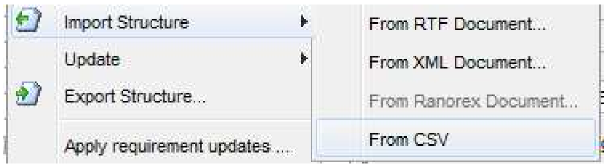


Figure 18 - Menu - Import Test Cases from Excel

When activating the menu Test Structure -> Import Structure -> From CSV Figure 19 is displayed. In that dialog you can assign Excel to TEMPPO attributes.

Mandatory attributes: Name and ID have to be assigned, otherwise an import cannot be performed.

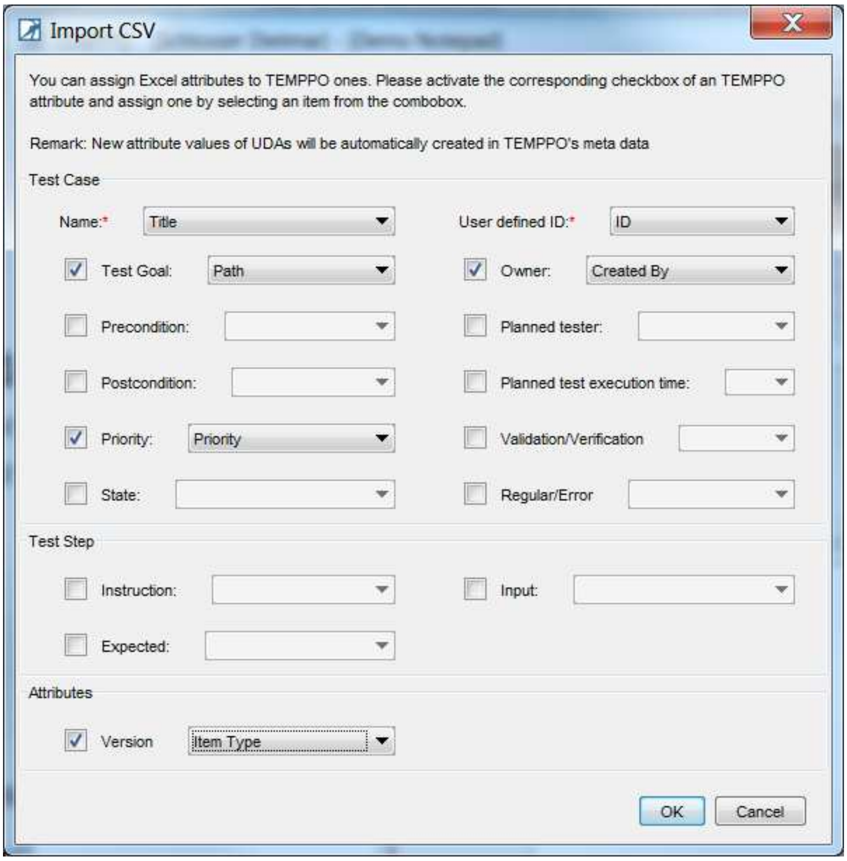
Fixed attributes like test goal, state…: can be assigned to any Excel attributes. Attribute values have to be identically. If they’re different, the default value will be set.

If the **owner** is selected and the user is not available in TEMPPO, the current importer is used as owner.

Test step: The 3 attributes Instruction, Expected, Input can be assigned with exactly 1 step.

User Defined Attributes: Any UDAs can be assigned with Excel attributes. If a value to be assigned doesn’t exist, it will be created automatically.

To assign the attributes above, please click the checkbox.



Import CSV

You can assign Excel attributes to TEMPPO ones. Please activate the corresponding checkbox of an TEMPPO attribute and assign one by selecting an item from the combobox.

Remark: New attribute values of UDAs will be automatically created in TEMPPO's meta data

Test Case

Name:* Title

User defined ID:* ID

☑ Test Goal: Path

☑ Owner: Created By

☐ Precondition:

☐ Planned tester:

☐ Postcondition:

☐ Planned test execution time:

☑ Priority: Priority

☐ Validation/Verification

☐ State:

☐ Regular/Error

Test Step

☐ Instruction:

☐ Input:

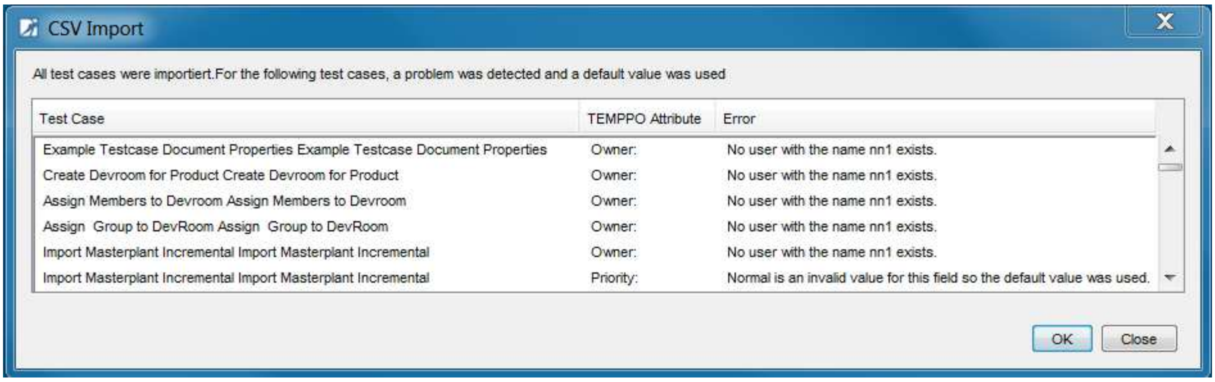
☐ Expected:

Attributes

☑ Version Item Type

Figure 19 - Assign Excel to TEMPPO attributes

After pressing OK the user is informed about errors of the import.



CSV Import

All test cases were importiert.For the following test cases, a problem was detected and a default value was used

Test Case	TEMPPO Attribute	Error
Example Testcase Document Properties Example Testcase Document Properties	Owner:	No user with the name nn1 exists.
Create Devroom for Product Create Devroom for Product	Owner:	No user with the name nn1 exists.
Assign Members to Devroom Assign Members to Devroom	Owner:	No user with the name nn1 exists.
Assign Group to DevRoom Assign Group to DevRoom	Owner:	No user with the name nn1 exists.
Import Masterplant Incremental Import Masterplant Incremental	Owner:	No user with the name nn1 exists.
Import Masterplant Incremental Import Masterplant Incremental	Priority:	Normal is an invalid value for this field so the default value was used.

Figure 20 - CSV Import Result

– Build manually
– XML-Import

3.1.2.3.4 Referring to requirement numbers or names

On the other hand there is the possibility to link requirements to test packages (as well as test cases). In some projects it might be necessary to track the requirements or rather their IDs from defining until executing test cases.

But before linking those to test packages it might be necessary to define and assign them for the project (see chapter 5.4.6).

In TEMPPO Requirement Manager you can create requirement structures. In TEMPPO Administrator requirement structures are assigned to TEMPPO projects. After that you can link the test cases to these requirements.

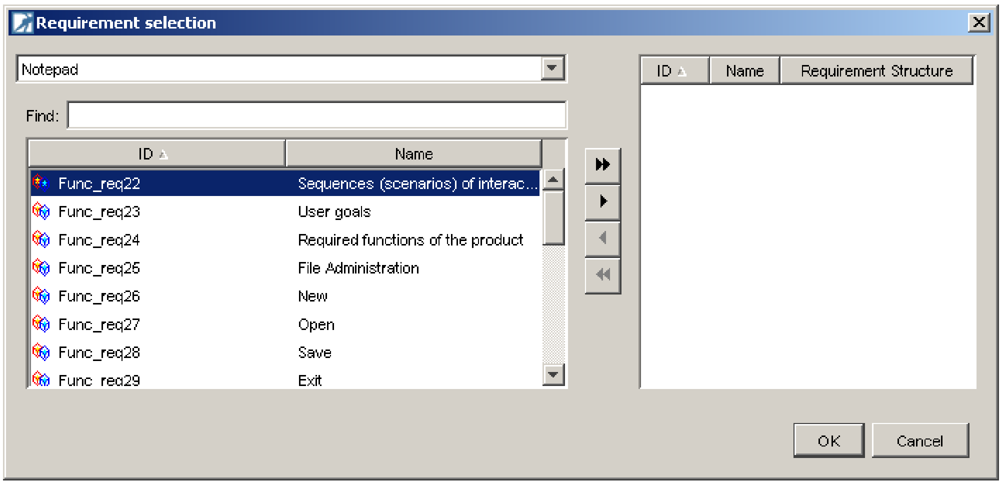
You have two possibilities to relate requirements to test packages or cases: **Add** and/or **inherit from parent test packages**.

3.1.2.3.4.1 Add Requirement

For adding a requirement you press the button **Add requirement** and Figure 31 is shown. At first you have to select the **requirement structure** and its requirements are listed. The requirement selection dialog allows you to link one, more or all requirement(s) to the package.

If more requirements are linked, they will be displayed in alphabetical order.

All requirements selected before are shown grey.



Notepad

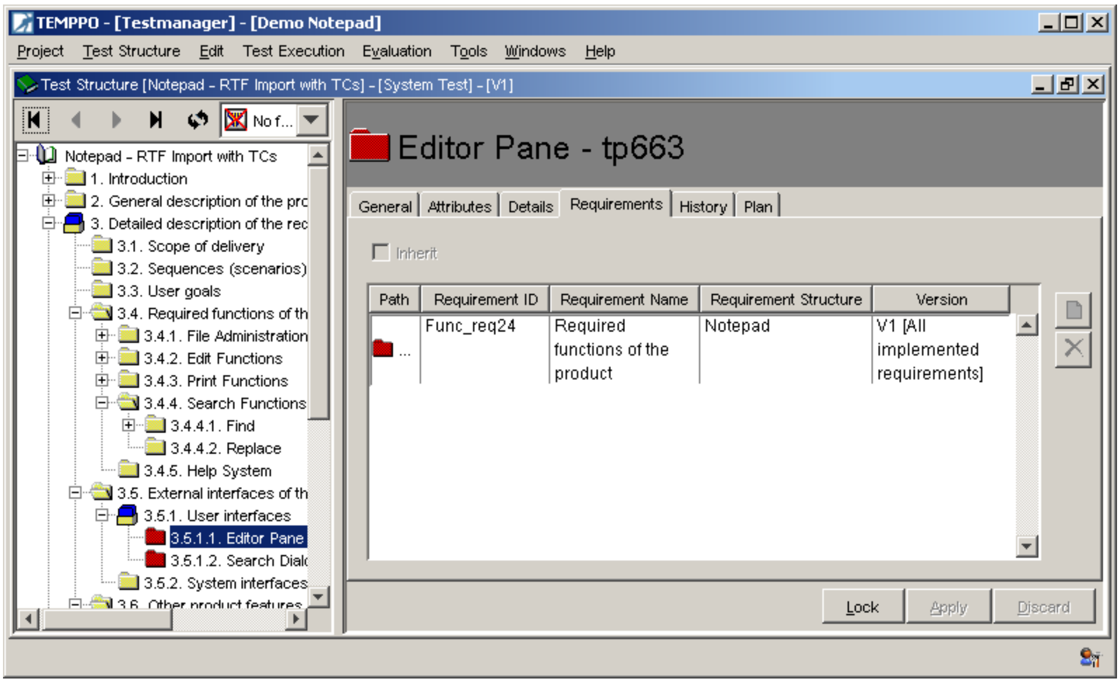
Find:

ID	Name
Func_req22	Sequences (scenarios) of interac...
Func_req23	User goals
Func_req24	Required functions of the product
Func_req25	File Administration
Func_req26	New
Func_req27	Open
Func_req28	Save
Func_req29	Exit

ID	Name	Requirement Structure

Figure 31 - Requirements: Selection

After pressing **OK** the requirement(s) are linked to the test package.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No f...
Notepad - RTF Import with TCs
1. Introduction
2. General description of the prc
3. Detailed description of the rec
3.1. Scope of delivery
3.2. Sequences (scenarios)
3.3. User goals
3.4. Required functions of th
3.4.1. File Administration
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of th
3.5.1. User interfaces
3.5.1.1. Editor Pane
3.5.1.2. Search Dialc
3.5.2. System interfaces
3.6. Other product features
Editor Pane - tp663
General
Attributes
Details
Requirements
History
Plan
Inherit
Lock
Apply
Discard

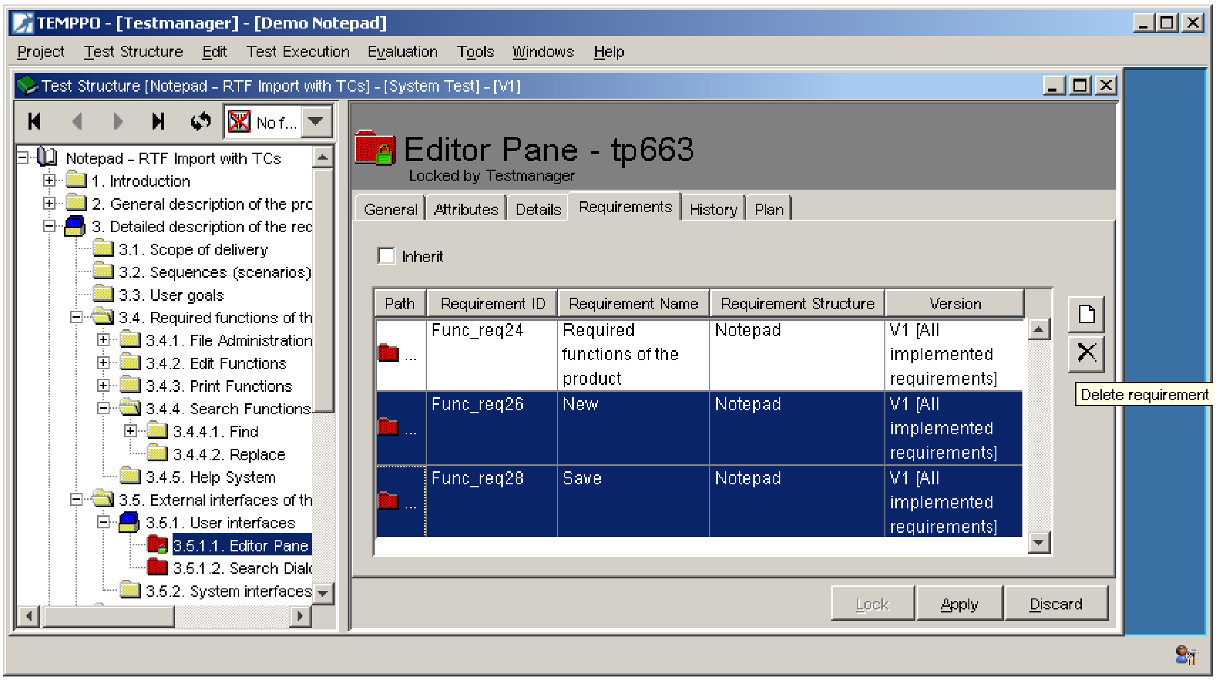
Path	Requirement ID	Requirement Name	Requirement Structure	Version
🟥 ...	Func_req24	Required functions of the product	Notepad	V1 [All implemented requirements]

Figure 32 - Test package - linked requirements

The button **Add requirement** is always enabled, if the test package is locked.

3.1.2.3.4.2 Delete requirement

If a test package is locked and at least one linked requirement is selected, the button **Delete requirement** is enabled.

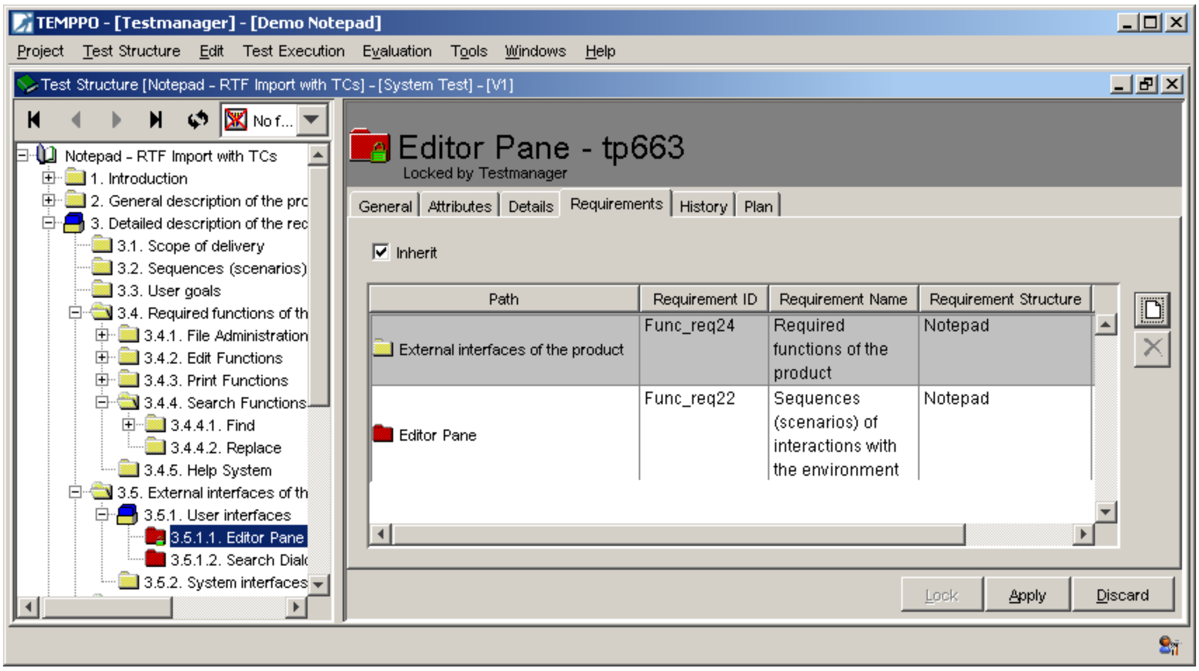

TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No f...
Notepad - RTF Import with TCs
1. Introduction
2. General description of the prc
3. Detailed description of the rec
3.1. Scope of delivery
3.2. Sequences (scenarios)
3.3. User goals
3.4. Required functions of th
3.4.1. File Administration
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of th
3.5.1. User interfaces
3.5.1.1. Editor Pane
3.5.1.2. Search Dialc
3.5.2. System interfaces
Editor Pane - tp663
Locked by Testmanager
General
Attributes
Details
Requirements
History
Plan
Inherit
Delete requirement
Lock
Apply
Discard

Path	Requirement ID	Requirement Name	Requirement Structure	Version
🟥 ...	Func_req24	Required functions of the product	Notepad	V1 [All implemented requirements]
🟥 ...	Func_req26	New	Notepad	V1 [All implemented requirements]
🟥 ...	Func_req28	Save	Notepad	V1 [All implemented requirements]

Figure 33 - Test Package - Delete requirement(s)

3.1.2.3.4.3 Inherit from parent test packages

If you associate requirements to test packages, we assume that the sub test package or children nodes may also refer to the same requirement. It would be very time consuming to relate always the same requirements as many times as you have children test packages. Therefore TEMPPO offers you to inherit requirements from parent test packages.



Editor Pane - tp663
Locked by Testmanager

☑ Inherit

Path	Requirement ID	Requirement Name	Requirement Structure
External interfaces of the product	Func_req24	Required functions of the product	Notepad
Editor Pane	Func_req22	Sequences (scenarios) of interactions with the environment	Notepad

Figure 34 – Requirement: Inherited

Figure 34 shows the selected test package **3.5.1.1 Editor Pane** and an activated checkbox **inherit**, which has the effect, that the test package inherits the requirements from its parent or its parent **External interfaces of the product**. Inherited requirements are displayed disabled so no selecting or deleting can be done. The inherited requirements are grouped by the test package names and within each group by the requirements in alphabetical order.

After that the enabled – not inherited – but also alphabetically ordered part is displayed (see Figure 34). If a user defines one requirement for two different levels within the inheritance tree both requirements are displayed within the requirement list. It is not possible to define one requirement more than once within one inheritance level.

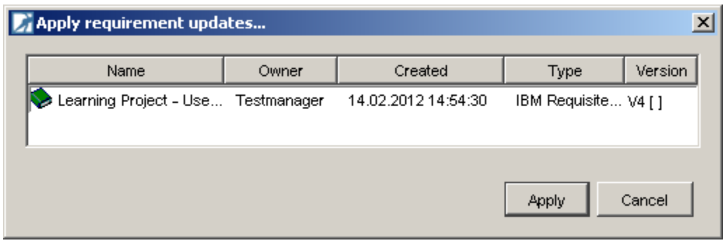
You might ask about the reason of linking the requirements to the tests and test packages. The answer is quite easy and will improve your whole process. On one hand you can analyze your test structure regarding to test coverage (e.g. how many test cases are related to the each requirement) and on the other hand you can make a statement about the quality of the SUT (e.g. how many bugs are related to a special requirement?).

3.1.2.3.4.4 Apply requirement updates

Requirement structures and test structures live in fully separated environments. So it can be that a latest test structure version works with a not latest requirement structure version. This is always the case after an update of a requirement structure. An automatic update of the current test structure version to the latest requirement structure version may not be wanted by the user and would cause problems anyway, e.g. if more than one user work on this version. It’s only possible to update to the latest requirement structure version.

So an update to the latest requirement structure version can only be made manually initiated. Open the TEMPPO test management environment and open the project and open the project and test structure. The menu item **Test**

Structure > Apply requirement updates… is only enabled if the main/LATEST version of the test structure is opened. By Pressing the menu **Test Structure > Apply requirement updates…** Figure 35 is shown. All assigned and updated requirement structures are shown in the list. Select the requirement structures when the assignments should be updated. By clicking **Apply**, the test structure is checked in and then propagation is executed.



Apply requirement updates...

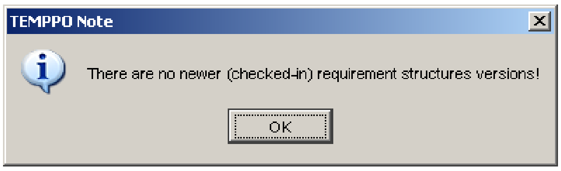
Name	Owner	Created	Type	Version
Learning Project - Use...	Testmanager	14.02.2012 14:54:30	IBM Requisite...	V4 [ ]

Figure 35 – Apply requirement updates

All links are changed to the new version of the requirements: Deleted requirements are unassigned from the test cases.

Updating a requirement structure does not cause that a task list is created, a task list (3.3.18) is created internally if the test structure is updated to the latest requirement structure version.

If there is no implicit propagation to the latest requirement structure version, Figure 36 is shown.



TEMPPO Note

There are no newer (checked-in) requirement structures versions!

Figure 36 – No requirement structure for updating

3.1.2.4 Designing Test Cases

Since there is an integration of TEMPPO Test Manager and TEMPPO Designer (IDATG) there are 2 possibilities for designing test cases. On the one hand the traditional one with TEMPPO Test Manager and the one with TEMPPO Designer (IDATG) (see chapter 7).

The last step of the design phase is the design of test cases. Note that test case design does not include the implementation (writing test steps or test scripts). The conceptual work of test case design has to be accomplished outside of TEMPPO. E.g. if you apply an equivalence partitioning technique, you will usually create a table with classes of input values and test cases covering different combinations of those. A unique name must be applied to the resulting test cases and they are ready to be inserted into the appropriate test package in the TEMPPO test structure.

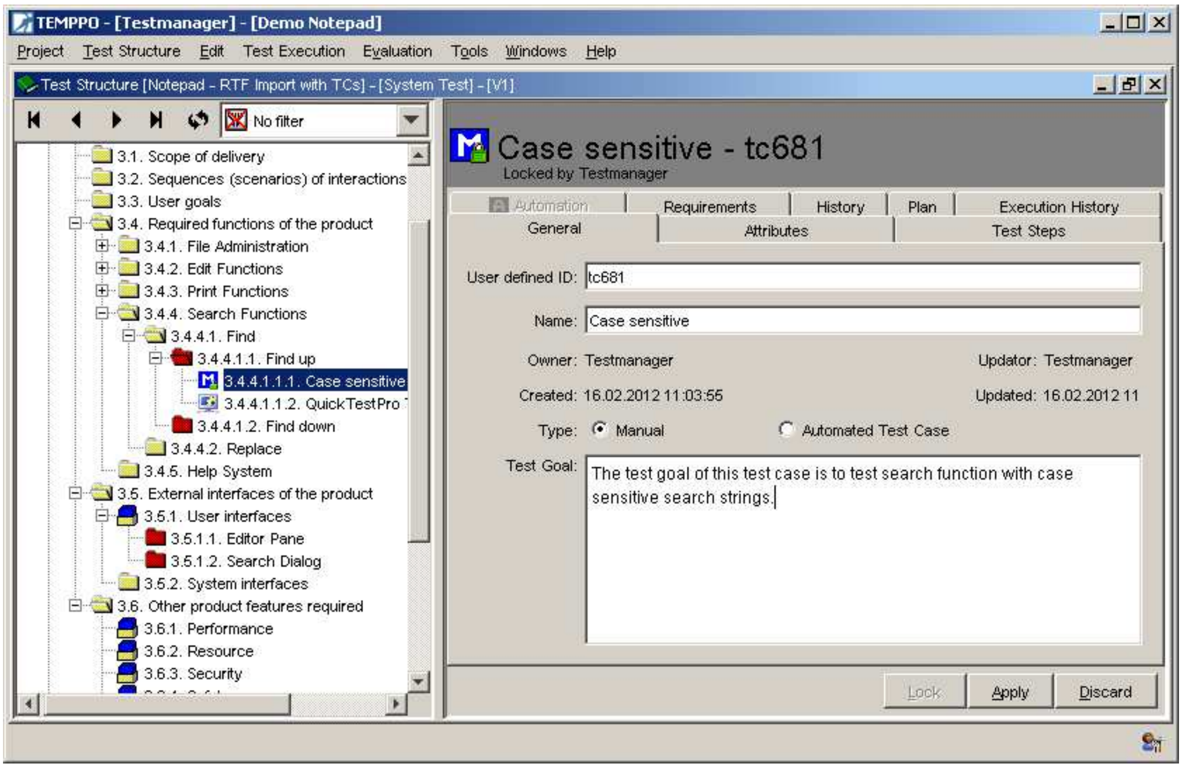
With the menu item **Test Structure > New > Test Case**, a new test case is created below the selected test package. This menu item is enabled, if a test structure is opened and a test package is selected. It can be activated by the menu or a right mouse click.

A new child node M (manual) or A (automated) is created below the selected one and a new name has to be entered. The default name is **New Test Case**.

The test cases tab consists of the tabs **General**, **Attributes**, **Preconditions** and **Postconditions**, **Test Steps**, **Automated**, **Requirements**, **History**, **Execution History** and **Plan**. During design you just use General, Attributes Requirements and Plan.

General:
You have to specify the **name**, **type** (either **manual** or **automated**) and the **goal** of the test case. The test case developer should be able to implement test steps based on this description. The automation tab is only enabled, in case of automated test cases.
Furthermore a **User Defined ID** is suggested by the system that can be edited. In many projects, requirements get unique IDs. It is recommended to use these IDs (plus some serial number) as test case IDs.

ⓘ **You can search for test cases by ID or name by with TEMPPO's find feature.**



User defined ID: tc681

Name: Case sensitive

Owner: Testmanager

Updator: Testmanager

Created: 16.02.2012 11:03:55

Updated: 16.02.2012 11

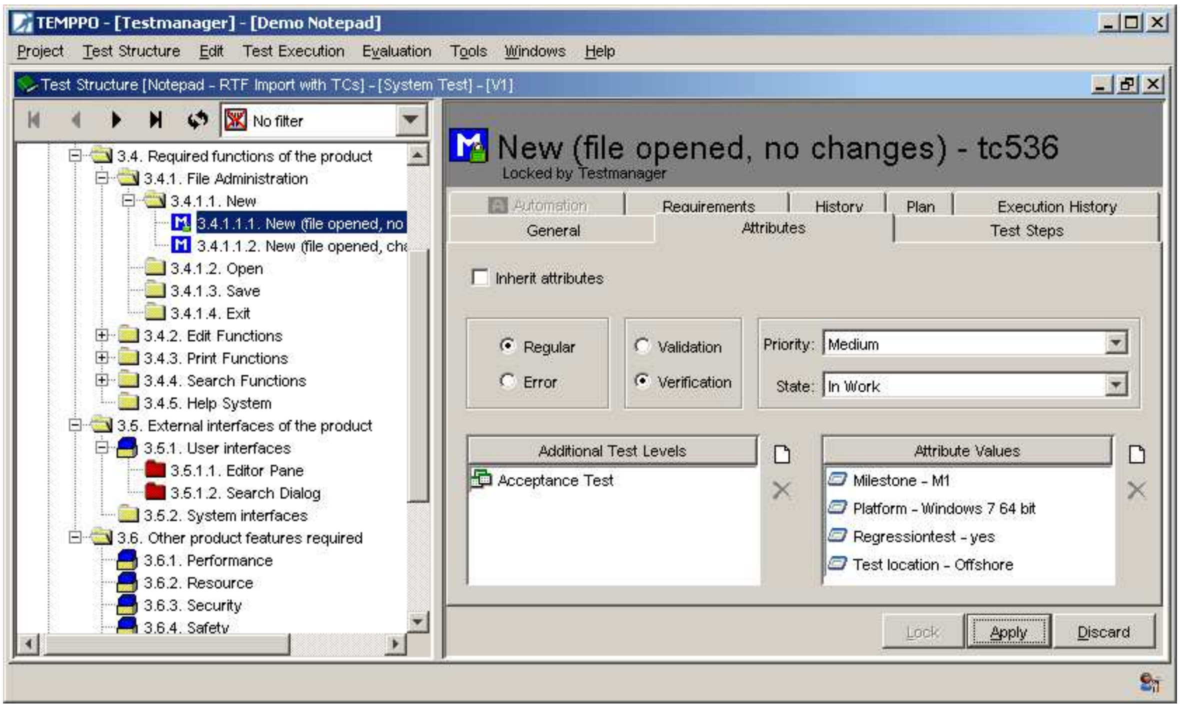
Type: ● Manual ○ Automated Test Case

Test Goal: The test goal of this test case is to test search function with case sensitive search strings.

Figure 37 – Manual Test Case, General

ⓘ **It is also possible to design automated test cases without having any script.**

Attributes



- 3.4. Required functions of the product
  - 3.4.1. File Administration
    - 3.4.1.1. New
      - 3.4.1.1.1. New (file opened, no
      - 3.4.1.1.2. New (file opened, cha
    - 3.4.1.2. Open
    - 3.4.1.3. Save
    - 3.4.1.4. Exit
  - 3.4.2. Edit Functions
  - 3.4.3. Print Functions
  - 3.4.4. Search Functions
  - 3.4.5. Help System
- 3.5. External interfaces of the product
  - 3.5.1. User interfaces
    - 3.5.1.1. Editor Pane
    - 3.5.1.2. Search Dialog
  - 3.5.2. System interfaces
- 3.6. Other product features required
  - 3.6.1. Performance
  - 3.6.2. Resource
  - 3.6.3. Security
  - 3.6.4. Safety

New (file opened, no changes) - tc536
Locked by Testmanager

☐ Inherit attributes

● Regular ○ Error

○ Validation ● Verification

Priority: Medium

State: In Work

Additional Test Levels	Attribute Values
Acceptance Test	Milestone - M1
	Platform - Windows 7 64 bit
	Regressiontest - yes
	Test location - Offshore

Figure 38 – Attributes

You have to specify, whether the test case is a **regular** one or tests an **error condition**. A regular test case pursues the aim to test a specified requirement. On the other hand an error case checks the application if e.g. a faulty input is rejected.

You may also select the purpose of test case execution: for **verification** or **validation**.

Furthermore you can select a value of the attribute **priority**, which mirrors the topic of risk oriented testing. Each test case is given a priority attribute, which reflects the test case’s importance either from the customer point of view or from the complexity point of view. It is important to note that risk oriented testing is no substitute for a complete run of all test cases. However it should be taken as a guideline telling which tests have to be performed prior to shipment even when time is short. After shipment the remaining tests still have to be carried out.

Suggestion for the definition of test case priority attributes (usually they should be defined in the test plan):

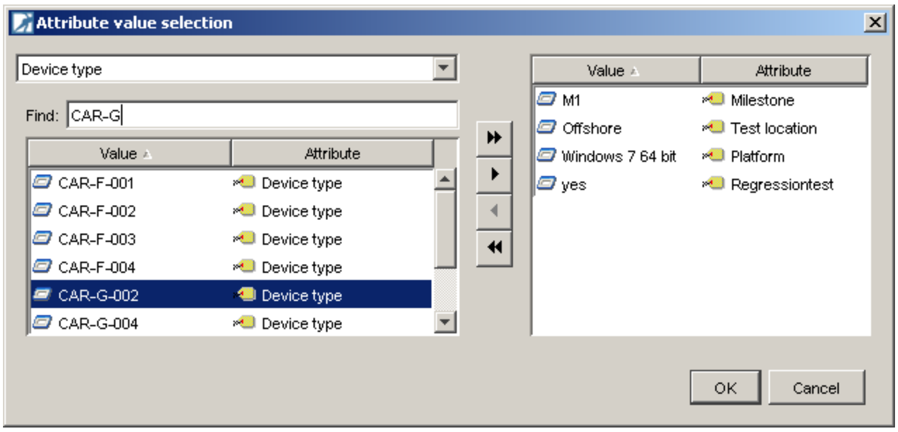
- **Top**: test case whose main concern is testing functionality. If this functionality is not implemented properly the customer may be disgruntled. Interrelationship to other functionality is very strong, such that system behavior may be concerned.
- **High**: test case whose main concern is testing an often used functionality. Therefore misbehavior would be immediately evident. Strong interrelationship to other functionality or complex algorithm involved.
- **Medium**: test case whose main concern is testing functionality that is not frequently used or not in the main interest of the customer. Therefore misbehavior is more likely to be “accepted” by the customer. Weak interrelationship to other functionality
- **Low**: test case whose main concern is testing functionality that is not frequently used or not in the main interest of the customer. Therefore misbehavior is more likely to be “accepted” by the customer. Complexity of underlying SW is small.

You also have to specify a working or development state for the test case. This is necessary for reporting, analysis, and creating test suites. For example, the test manager wants to create a test suite only with test cases that are ready for test execution. If such an attribute does not exist or has a wrong value, the test execution will fail. The working state consists of the following values:

- **Designed**: The test case is created and the data in the tabs **General**, **Attributes**, **Requirements** are specified. This state was introduced for the purpose of supporting test teams that separate the work into designing and implementing test cases. This is the default state of a newly created test case.
- **In Work**: The test case already contains test steps, but is not ready implemented
- **Ready for Review**: This state is not obligatory; it can be used for projects, where test specification reviews are mandatory.
- **Ready**: The test case is already implemented, reviewed and can be used for test case execution.
- **Approved**: This state is optional and mostly used for automated test cases. Sometimes a test case is ready for execution, but fails due to errors in the test script. If a test case is executable as specified in its test steps / test script, it gets the state approved.

Additional test levels can be defined for using one test structure for different test levels. For example if you have a test case for the standard test level "System Test" (defined when creating the test structure, see 3.1.2.2.1) and want to use the test case in a test execution for an acceptance test, you can achieve this by adding **Acceptance Test** as an additional test level for this test case. For the execution of the acceptance test, you would then specify this test level as a filter criterion for your test suite (see 3.1.4.1).
Additionally you can set attributes specified in the **Meta Data Editor** (see **5.3**), which can also be used for creating a test suite.

By specifying a sub-string in the **Find** field you can easily navigate to the attribute(s) to be related to the test case. Multi select allows "moving" several values to the right part of the window (see Figure 39).



Device type

Find: CAR-G

Value	Attribute
CAR-F-001	Device type
CAR-F-002	Device type
CAR-F-003	Device type
CAR-F-004	Device type
CAR-G-002	Device type
CAR-G-004	Device type

Value	Attribute
M1	Milestone
Offshore	Test location
Windows 7 64 bit	Platform
yes	Regressiontest

Figure 39 – Attribute value selection

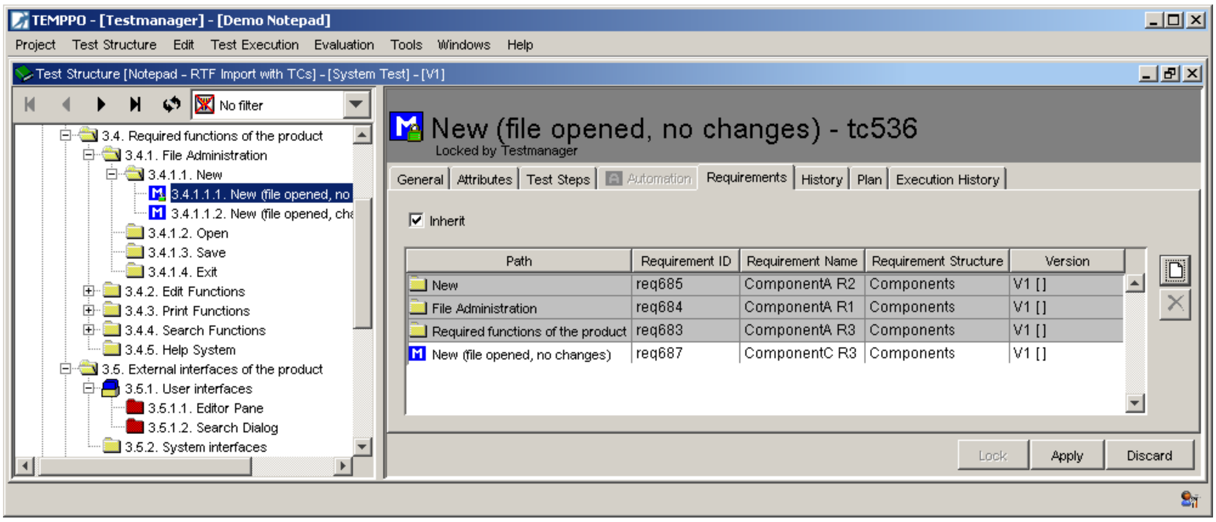
The test case design and the test design phase as a whole are finished when the test cases are designed and obsolete test packages are removed (do not forget to do that).

The tasks of designing and developing (implementation phase) test cases are sometimes overlapping, especially if designer and implementer is the same person or you import a complete test structure (see 3.1.2.2.4.2).

- **Requirement**

Optionally you can use the feature requirement tracking. As described in chapter 3.1.2.3.4, requirements can be associated to test packages and inherited. Of course it is possible to inherit requirements from test packages and/or relate requirements to test cases.

Figure 40 shows the test case **3.4.1.1.1 New (file opened, no changes)**, which inherits from the test package 3.4 Required functions of the product the requirements `ComponentA R3` inherits from the test package **3.4.1.1 New** the requirement `ComponentA R2` has a link to the requirement `ComponentC R3`



Path	Requirement ID	Requirement Name	Requirement Structure	Version
New	req685	ComponentA R2	Components	V1 []
File Administration	req684	ComponentA R1	Components	V1 []
Required functions of the product	req683	ComponentA R3	Components	V1 []
New (file opened, no changes)	req687	ComponentC R3	Components	V1 []

Figure 40 - Requirement: Test case

You might ask for reason of the requirement tracking. On one hand you can analyze the test structure and on the other hand such a requirement analysis can be applied to the test suite. For test structures you can create an analysis about the quality of your test coverage. For test suites the quality of requirements can be visualized.

For detailed description of requirement analysis see chapter 3.1.5.2.

- **History**

History entries are generated for test packages and test cases. The entries are shown in an own tab **History**. Only the entries until the last check-in are shown. If an item is merged, the history is deleted and one new entry is made (e.g.: All, V35.1.1., merge).

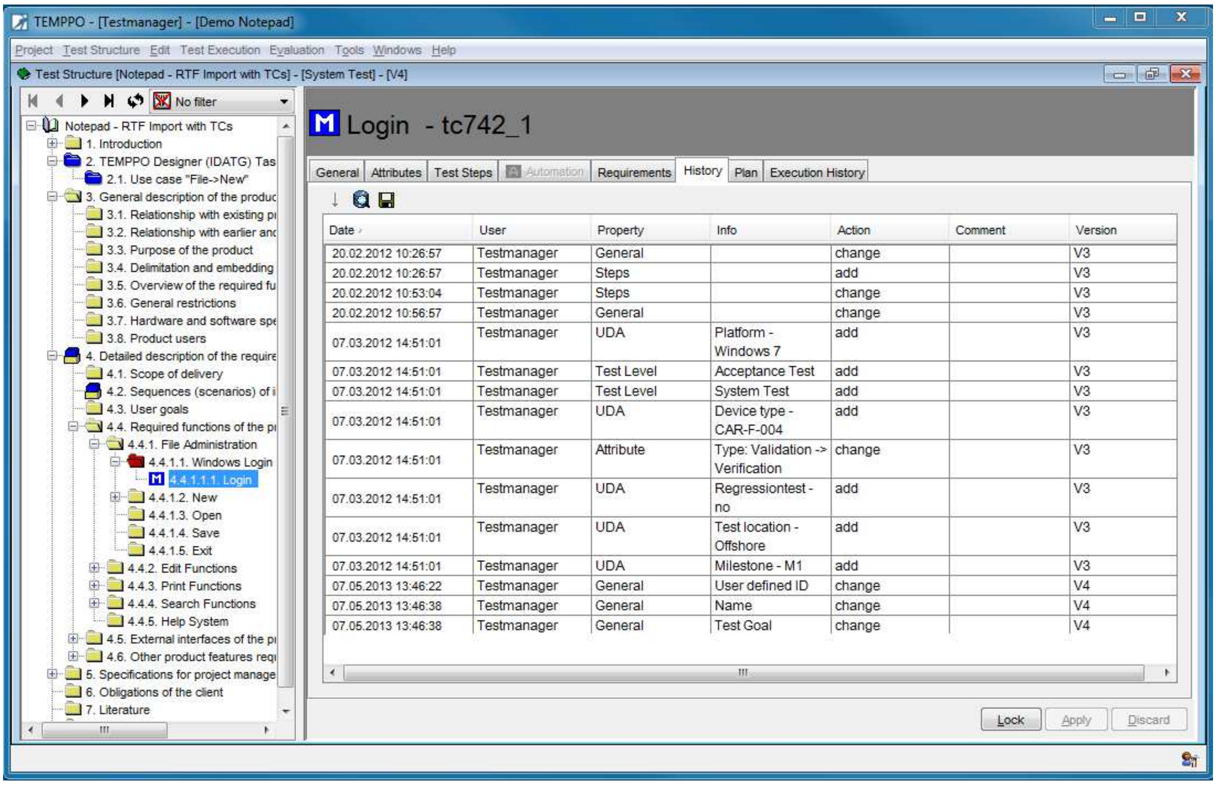
The tab has the following columns:

- **Date**: changed on
- **User**: changed by
- **Property**: possible values: Requirement, UDA, Test Level, Steps, General, All (if item is merged)
- **Name**: Name of Property; when Property is in (General, Steps), name = "–", when Property is (All), name is Version of merged Item
- **Action**: possible values: add, change, delete, merge
- **Info**: additional comment for changing reason

- **Version**: of history entry. With button **Display all entries** all history entries of all test structures can be loaded

Action is depending on Property (possible values):

- **Requirement**: add, delete
- **UDA**: add, delete
- **Test Level**: add, delete
- **Steps**: add, change, delete
- **General**: change; **Info** = Name, User defined ID or Test goal
- **All**: merge


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V4]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. TEMPPO Designer (IDATG) Tas
2.1. Use case "File->New"
3. General description of the produc
3.1. Relationship with existing pı
3.2. Relationship with earlier anc
3.3. Purpose of the product
3.4. Delimitation and embedding
3.5. Overview of the required fu
3.6. General restrictions
3.7. Hardware and software spe
3.8. Product users
4. Detailed description of the require
4.1. Scope of delivery
4.2. Sequences (scenarios) of i
4.3. User goals
4.4. Required functions of the pı
4.4.1. File Administration
4.4.1.1. Windows Login
4.4.1.1.1. Login
4.4.1.2. New
4.4.1.3. Open
4.4.1.4. Save
4.4.1.5. Exit
4.4.2. Edit Functions
4.4.3. Print Functions
4.4.4. Search Functions
4.4.5. Help System
4.5. External interfaces of the pı
4.6. Other product features requ
5. Specifications for project manage
6. Obligations of the client
7. Literature
M Login - tc742_1
General
Attributes
Test Steps
Automation
Requirements
History
Plan
Execution History
Lock
Apply
Discard

Date	User	Property	Info	Action	Comment	Version
20.02.2012 10:26:57	Testmanager	General		change		V3
20.02.2012 10:26:57	Testmanager	Steps		add		V3
20.02.2012 10:53:04	Testmanager	Steps		change		V3
20.02.2012 10:56:57	Testmanager	General		change		V3
07.03.2012 14:51:01	Testmanager	UDA	Platform - Windows 7	add		V3
07.03.2012 14:51:01	Testmanager	Test Level	Acceptance Test	add		V3
07.03.2012 14:51:01	Testmanager	Test Level	System Test	add		V3
07.03.2012 14:51:01	Testmanager	UDA	Device type - CAR-F-004	add		V3
07.03.2012 14:51:01	Testmanager	Attribute	Type: Validation -> Verification	change		V3
07.03.2012 14:51:01	Testmanager	UDA	Regressiontest - no	add		V3
07.03.2012 14:51:01	Testmanager	UDA	Test location - Offshore	add		V3
07.03.2012 14:51:01	Testmanager	UDA	Milestone - M1	add		V3
07.05.2013 13:46:22	Testmanager	General	User defined ID	change		V4
07.05.2013 13:46:38	Testmanager	General	Name	change		V4
07.05.2013 13:46:38	Testmanager	General	Test Goal	change		V4

Figure 41 – History tab

Within current test structure version, the number of history entries loaded (x) can be configured in settings (see 3.3.20.2.4). If there are more history entries, the button **Search next** is still enabled und the next x lines could be loaded.

It also possible to view all history entries of all test structure versions by pressing the button **Display all entries** and save them to CSV file.

The additional comment (“**Info**”) can be configured to be mandatory in **TEMPPO Administrator -> Project properties**. For each project this mandatory flag can be set.
If the mandatory flag is set on, a history comment is popped up on apply (Figure 42).

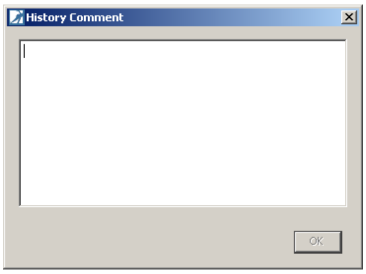


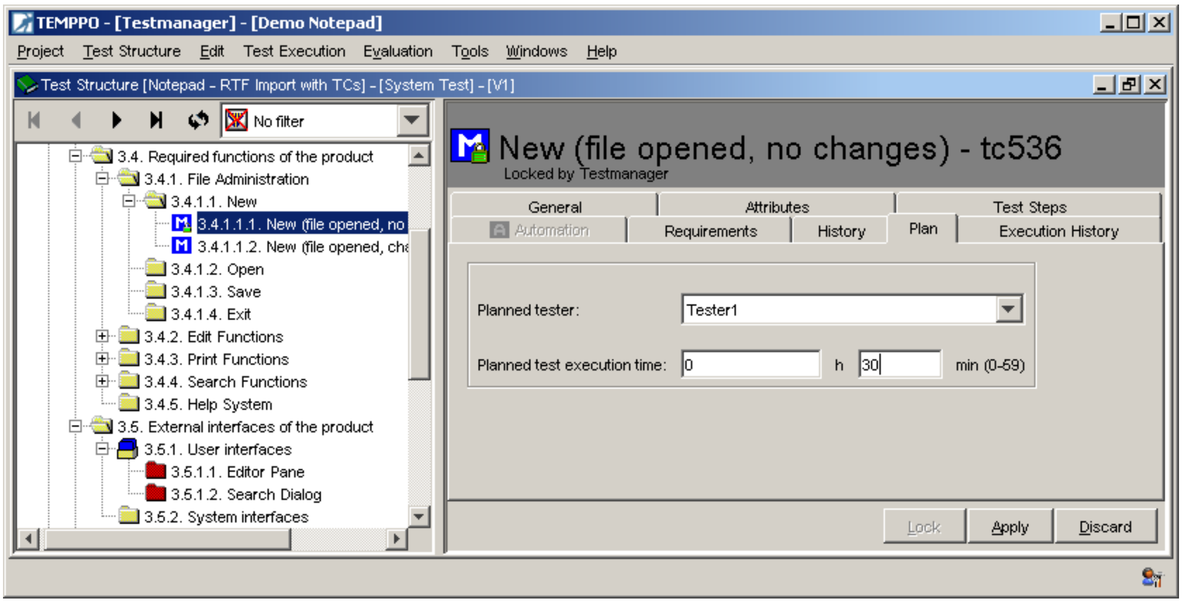
Figure 42 – History comment

- **Plan**

Since testing is a project within a development project TEMPPO offers possibilities for test project planning. TEMPPO do not wants to replace project planning tools like MS Project but presents features for resource and test execution planning.

Test case designers or test managers have also the task to estimate the execution effort for each test case as well as to appoint a tester (who executes it). The default value for the **Planned Tester** is the creator.

ⓘ **Do not estimate the effort for the test cases which passes! Most time testers spend their time on bug analysis! Therefore add that time also to the estimation.**

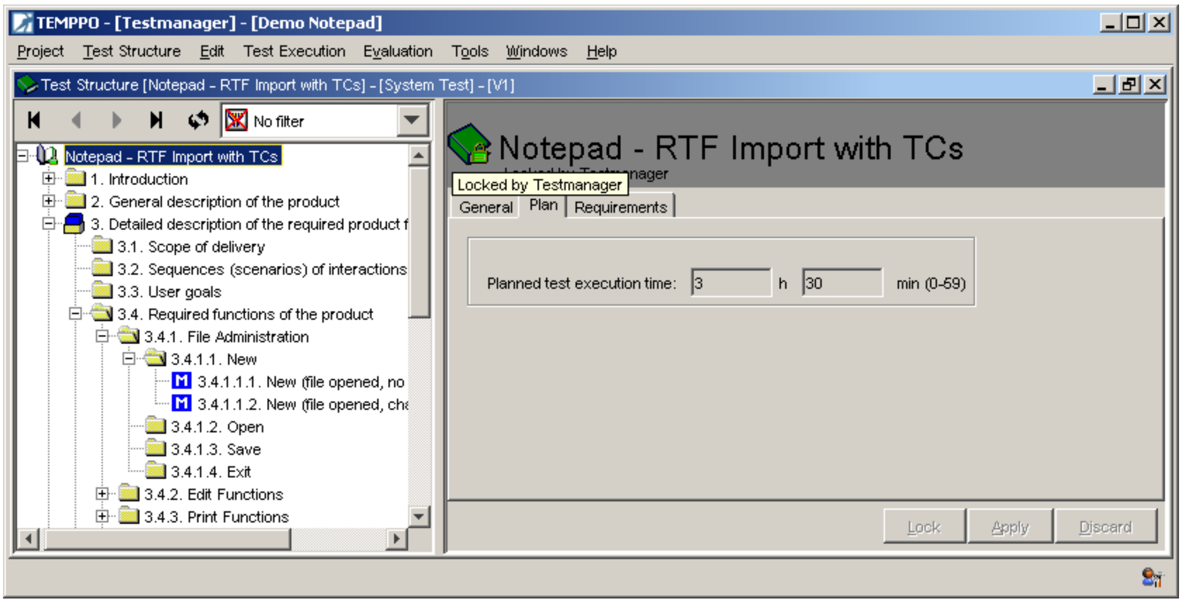


Planned tester: Tester1

Planned test execution time: 0 h 30 min (0-59)

Figure 43 - Tab Plan

For each test case the execution effort is entered in the format of hours and/or minutes. On test package and test structure level the values are displayed accumulated. After designing all test cases test manager clicks on test structure root, select tab **Plan** and gets immediately the estimated execution effort for all test cases (see Figure 44).



TEMPPO - [Testmanager] - [Demo Notepad]

Project Test Structure Edit Test Execution Evaluation Tools Windows Help

Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]

No filter

- Notepad - RTF Import with TCs
  - 1. Introduction
  - 2. General description of the product
  - 3. Detailed description of the required product f
    - 3.1. Scope of delivery
    - 3.2. Sequences (scenarios) of interactions
    - 3.3. User goals
    - 3.4. Required functions of the product
      - 3.4.1. File Administration
        - 3.4.1.1. New
          - 3.4.1.1.1. New (file opened, no
          - 3.4.1.1.2. New (file opened, cha
        - 3.4.1.2. Open
        - 3.4.1.3. Save
        - 3.4.1.4. Exit
      - 3.4.2. Edit Functions
      - 3.4.3. Print Functions

Notepad - RTF Import with TCs

Locked by Testmanager

General | Plan | Requirements

Planned test execution time: 3 h 30 min (0-59)

Lock Apply Discard

Figure 44 - Planned execution time for whole test structure

Additionally the designer enters a **Planned tester** for each test case who is planned to execute it. For **Analysis** of test structure **Planned Tester** is available to view whereas the analysis of different **Test execution efforts** makes no sense.

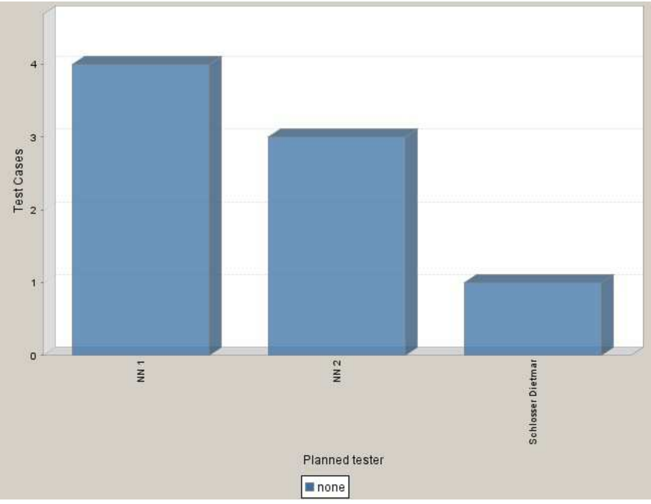


Figure 45 - Chart "Planned Tester"

Planned test execution time and Planned Tester can also be used for test structure filter and the logical expression of a test suite.

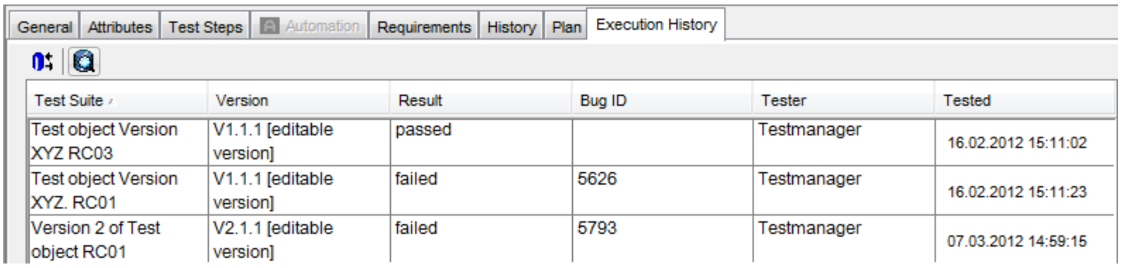
Planned test execution time and **Planned Tester** can also be reported for test cases, whereas for test packages only **Planned test execution time** is applicable.

Both attributes can be viewed in test suite, that offers the possibility to enter an **Actual test execution time** which can be applied to latest version as a learned lesson. For details see chapter 3.1.4.1.5.2.

- **Execution History**

This tab shows only executed results over the complete version tree of the test structure. For each test suite the following attributes are displayed:

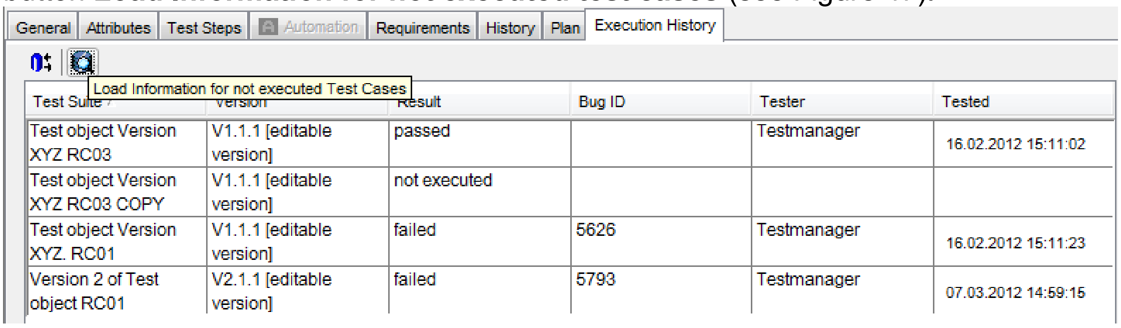
Test Suite, Version, Result, Bug ID, Tester, Tested



Test Suite	Version	Result	Bug ID	Tester	Tested
Test object Version XYZ RC03	V1.1.1 [editable version]	passed		Testmanager	16.02.2012 15:11:02
Test object Version XYZ. RC01	V1.1.1 [editable version]	failed	5626	Testmanager	16.02.2012 15:11:23
Version 2 of Test object RC01	V2.1.1 [editable version]	failed	5793	Testmanager	07.03.2012 14:59:15

Figure 46 - Execution History (only executed results)

If you want see all information for this test case over all test suites, you have to activate the button **Load Information for not executed test cases** (see Figure 47).



Test Suite	Version	Result	Bug ID	Tester	Tested
Test object Version XYZ RC03	V1.1.1 [editable version]	passed		Testmanager	16.02.2012 15:11:02
Test object Version XYZ RC03 COPY	V1.1.1 [editable version]	not executed			
Test object Version XYZ. RC01	V1.1.1 [editable version]	failed	5626	Testmanager	16.02.2012 15:11:23
Version 2 of Test object RC01	V2.1.1 [editable version]	failed	5793	Testmanager	07.03.2012 14:59:15

Figure 47 - Execution History (executed and not executed)

3.1.3 Test Case Implementation

The implementation phase covers the tasks of

- Implementing Test Steps
- Reworking the Test Structure
- implementing test frameworks (not covered by TEMPPO Test Manager)
- implementing test scripts for automation (not covered by TEMPPO Test Manager)

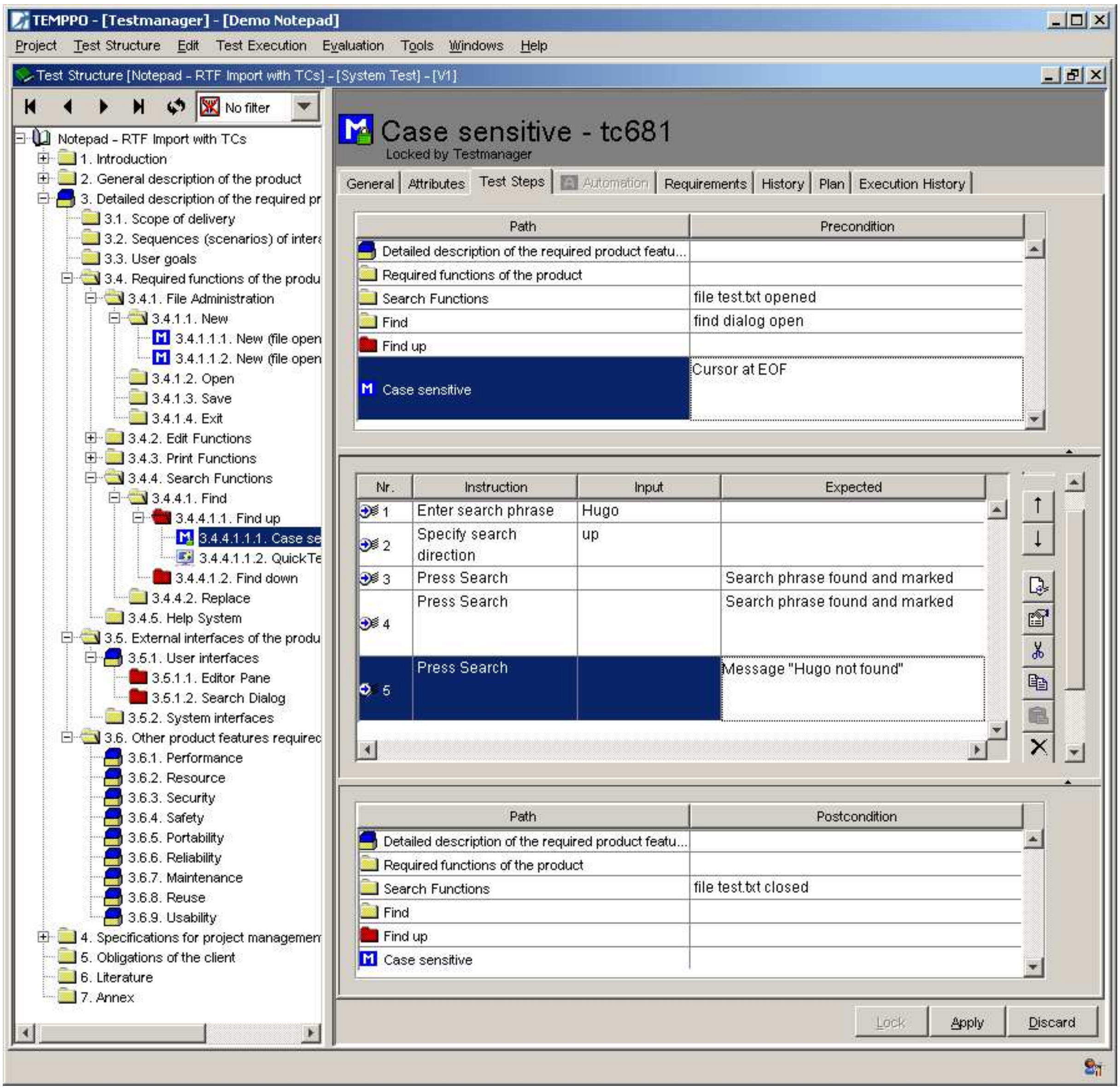
3.1.3.1 Implementing Test Steps

You continue your work in the tabs Test Steps and Automation of the test case tab.

3.1.3.1.1 Preconditions and Postconditions

You have to consider, which preconditions must be fulfilled before the test steps can be carried out during test execution. You specify this textually in the precondition table of the tab test steps. Additionally you can see the preconditions defined in the parent-test packages.

Additionally, you can also define postconditions, which mut be fulfilled after test steps execution. You specify this textually in the postcondition table of the tab test steps. You can see the postconditions defined in the parent-test packages, too.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required pr
3.1. Scope of delivery
3.2. Sequences (scenarios) of intere
3.3. User goals
3.4. Required functions of the produ
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file open
3.4.1.1.2. New (file open
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case se
3.4.4.1.1.2. QuickTe
3.4.4.1.2. Find down
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of the produ
3.5.1. User interfaces
3.5.1.1. Editor Pane
3.5.1.2. Search Dialog
3.5.2. System interfaces
3.6. Other product features requirec
3.6.1. Performance
3.6.2. Resource
3.6.3. Security
3.6.4. Safety
3.6.5. Portability
3.6.6. Reliability
3.6.7. Maintenance
3.6.8. Reuse
3.6.9. Usability
4. Specifications for project managemen
5. Obligations of the client
6. Literature
7. Annex
Case sensitive - tc681
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
Lock Apply Discard

Path	Precondition
Detailed description of the required product featu...	
Required functions of the product	
Search Functions	file test.txt opened
Find	find dialog open
Find up	
Case sensitive	Cursor at EOF

Nr.	Instruction	Input	Expected
1	Enter search phrase	Hugo	
2	Specify search direction	up	
3	Press Search		Search phrase found and marked
4	Press Search		Search phrase found and marked
5	Press Search		Message "Hugo not found"

Path	Postcondition
Detailed description of the required product featu...	
Required functions of the product	
Search Functions	file test.txt closed
Find	
Find up	
Case sensitive	

Figure 48 - Test Case, Preconditions and Postconditions

3.1.3.1.2Test Steps

The test step editor provides functions for creating, moving, cutting, copying and pasting one or more test steps within a test case or test structure wide to other test cases. The column layout can be individually adapted by moving and hiding columns.

When pressing P+ (**show preconditions**) above the Test steps-Edit-Buttons all the preconditions from the test case and the test packages above are displayed (see Figure 49). If the first test steps of a test case are implemented, the preconditions can be hidden by pressing P-.

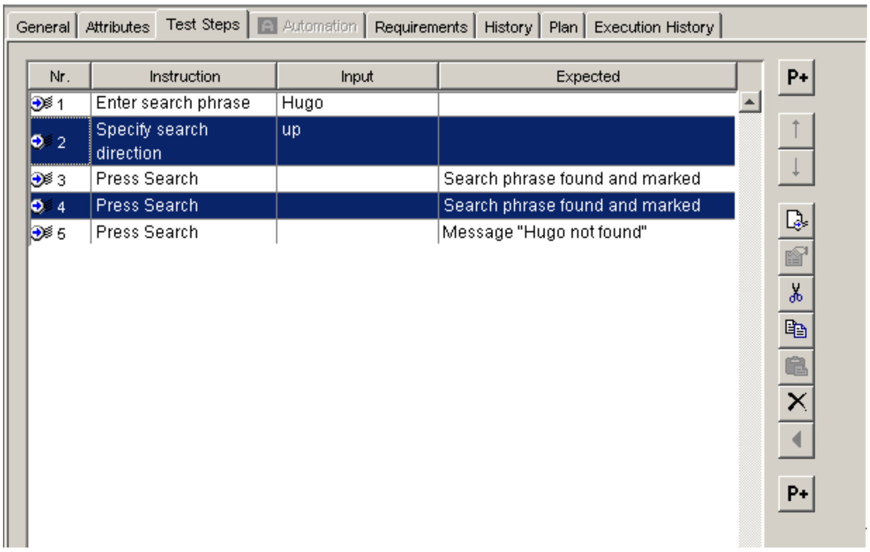
When pressing P+ (**show preconditions**) beneath the Test steps-Edit-Buttons all the postconditions from the test case and the test packages beneath are displayed (see Figure 49). If the first test steps of a test case are implemented, the postconditions can be hided by pressing P-.

When pressing , a new line for **Instruction**, **Input** and **Expected** (result) is inserted. The cell height is always 3 lines during editing. When pressing `Apply` the cell height is adapted to the whole line. The default cell height can be

configured in TEMPPO settings.
The test step number is created automatically.

With the two arrows (↓ , ↑) you can reorder steps by selecting a step and pressing the proper button. The test step numbers are updated automatically.

Test steps can be deleted by selecting them and pressing ✕.



Nr.	Instruction	Input	Expected
1	Enter search phrase	Hugo	
2	Specify search direction	up	
3	Press Search		Search phrase found and marked
4	Press Search		Search phrase found and marked
5	Press Search		Message "Hugo not found"

Figure 49 - Test Case, Test Step Editor

With **cut, copy, paste** you copy or move one or more test steps (via multi select) either within a test case or to other test cases in the test structure. By selecting a test step and pressing the corresponding button you apply these functions.

Even the short cuts <Ctrl>+X, <Ctrl>+C, and <Ctrl>+V are supported.

If the mouse cursor is located over the column headings, a menu for **hiding** and **displaying** the columns is popped up when pressing the right mouse button.

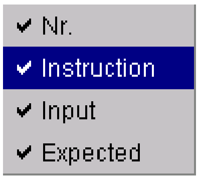
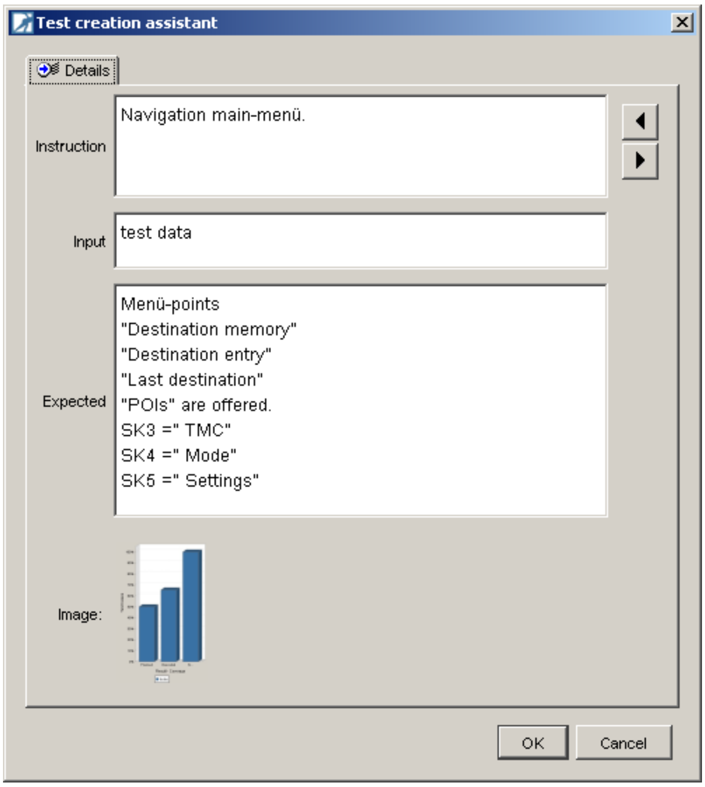


Figure 50 – Context menu for column hiding/displaying

3.1.3.1.3Test Creation Assistant

When pressing 📝 or double clicking a test step Figure 51 is displayed. The Test Creation Assistant can especially be used to edit text fields with a long description and navigate up and down between the test steps.

If the last step is opened and you press **Navigate down test step**, a new step is inserted.



Instruction: Navigation main-menü.

Input: test data

Expected: Menü-points
"Destination memory"
"Destination entry"
"Last destination"
"POIs" are offered.
SK3 =" TMC"
SK4 =" Mode"
SK5 =" Settings"

Image:

Figure 51 - Test Creation Assistant

3.1.3.1.4 Keyboard edit mode

Quick editing and navigation in the test step table is supported by:

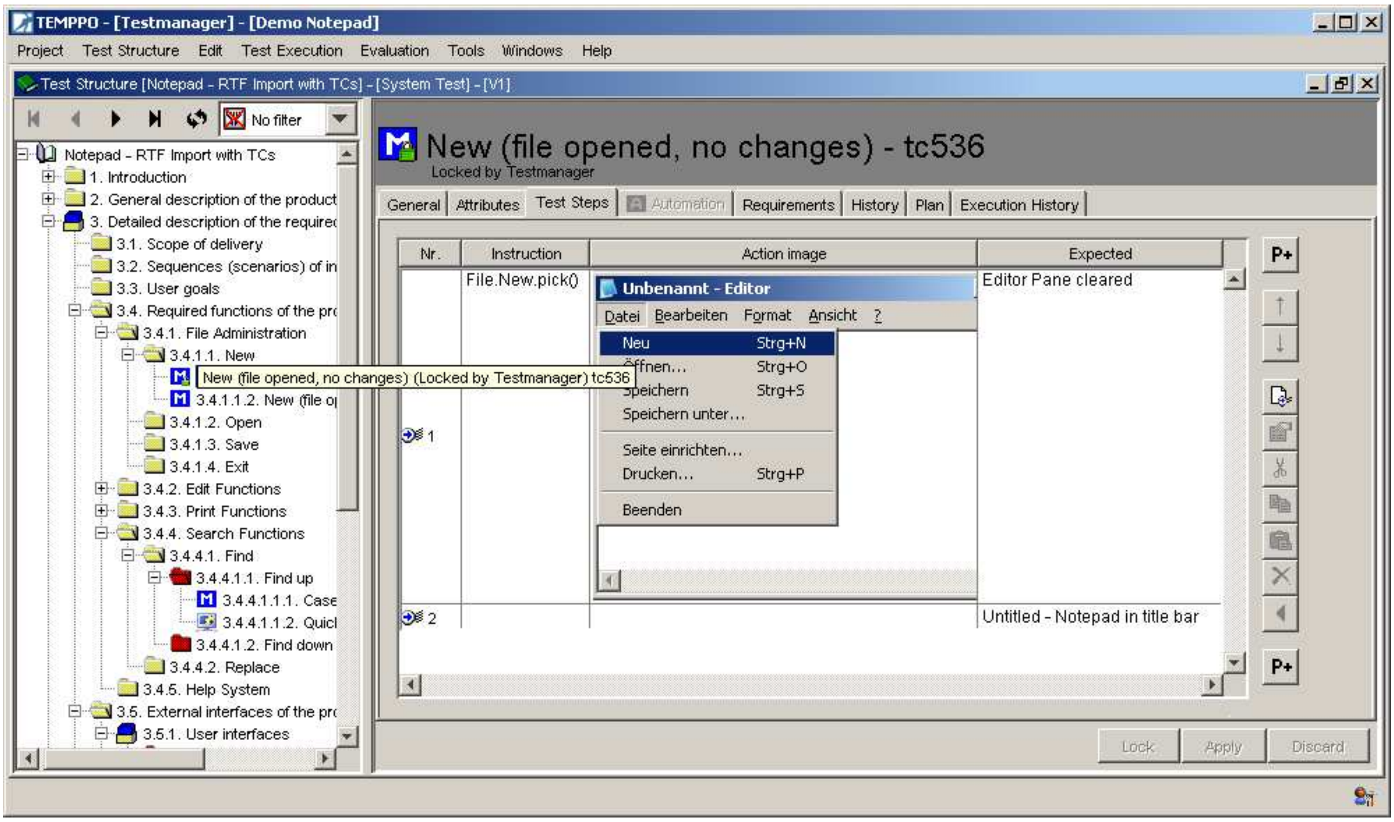
<F2>	start/stop edit mode of current table cell
<ESC>	stop edit mode, but changes are lost!
<TAB>, <Shift>+<TAB>	jump to next/previous table cell (only if not in edit mode)
↓, ↑	move one test step down / up
<Ctrl>+<TAB>	leave test step table

So if you want to edit test steps without using the mouse, you have to proceed like this:

<F2>	start edit mode
	Make your changes in the current cell
<F2>	stop edit mode
<TAB>	jump to next table cell
<F2>	start edit mode
...	and so on

3.1.3.1.5 Adding more columns

For creating test steps you have 3 fixed columns: **Instruction**, **Input** and **Expected Result**. In test projects it is sometimes necessary to add more columns for test steps.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the requirec
3.1. Scope of delivery
3.2. Sequences (scenarios) of in
3.3. User goals
3.4. Required functions of the prc
3.4.1. File Administration
3.4.1.1. New
New (file opened, no changes) (Locked by Testmanager) tc536
3.4.1.1.2. New (file o|
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case
3.4.4.1.1.2. Quicl
3.4.4.1.2. Find down
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of the prc
3.5.1. User interfaces
New (file opened, no changes) - tc536
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
Unbenannt - Editor
Datei Bearbeiten Format Ansicht ?
Neu Strg+N
Öffnen... Strg+O
Speichern Strg+S
Speichern unter...
Seite einrichten...
Drucken... Strg+P
Beenden
P+
P+
Lock Apply Discard

Nr.	Instruction	Action image	Expected
1	File.New.pick()		Editor Pane cleared
2			Untitled - Notepad in title bar

Figure 52 - New Columns in test steps

There are 2 possibilities to add columns:

1. TEMPPO Administrator, Tab **Metadata**, **User Fields** (See chapter 5.4.7.
   You can create new **Uploads** directly in the database.
   You have the possibility to define text or image columns.
2. Import XML

   On the other hand you may have an XML file (created by TEMPPO Designer (IDATG) or yourself), that you want to import to TEMPPO. If you follow the DTD (can be found in your XML directory), you can import test cases with additional columns.

   The definitions in tab **MetaData**, **User Fields** (TEMPPO Administrator application) are automatically created.

   An example of such an XML file is shown below:

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<!DOCTYPE TEMPPO_EXCHANGE SYSTEM "Test-structure.dtd">
<TEMPPO_EXCHANGE>
    <TP name="HumanResources">
            <TC name="Search_T1" test_goal="Transition Test" situation="Error">
                    <STEP instruction="" input="" expected="">
                            <USR_FIELD name="Actor" value="Sean Connery"/>
                            <USR_FIELD name="ExecutionRequest" value=""/>
                            <USR_FIELD name="Data" value="line 1
                                    line 2
                                    line 3"/>
                    </STEP>
                    <STEP instruction="" input="" expected="">
                            <USR_FIELD name="Actor" value="Sean Connery"/>
                            <USR_FIELD name="ExecutionRequest" value=""/>
                            <USR_FIELD name="Data" value="line 1
                                    line 2
                                    line 3"/>
                    </STEP>
            </TC>
      </TP>
</TEMPPO_EXCHANGE>
```

3.1.3.1.6 Upload / paste / reference images

For adding pictures to such a column you have to select such a cell, press the right mouse button and activate the menu **Upload**, **Delete**, **Paste, Copy** or **Reference**.

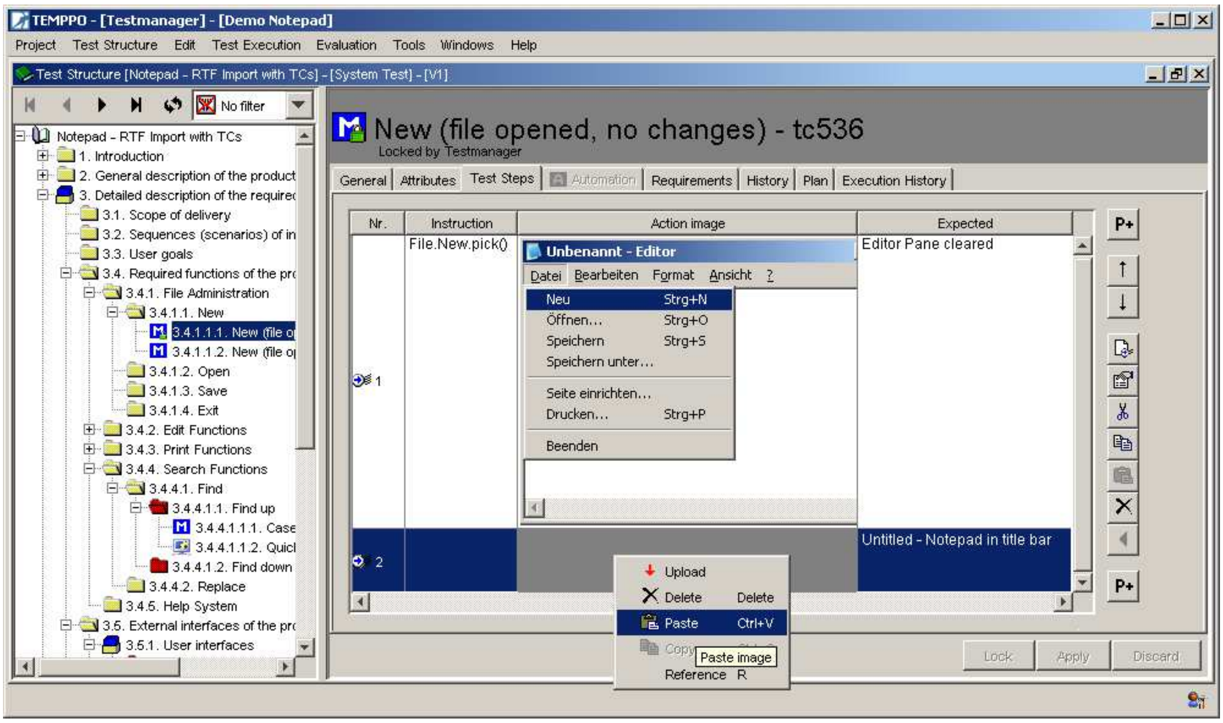
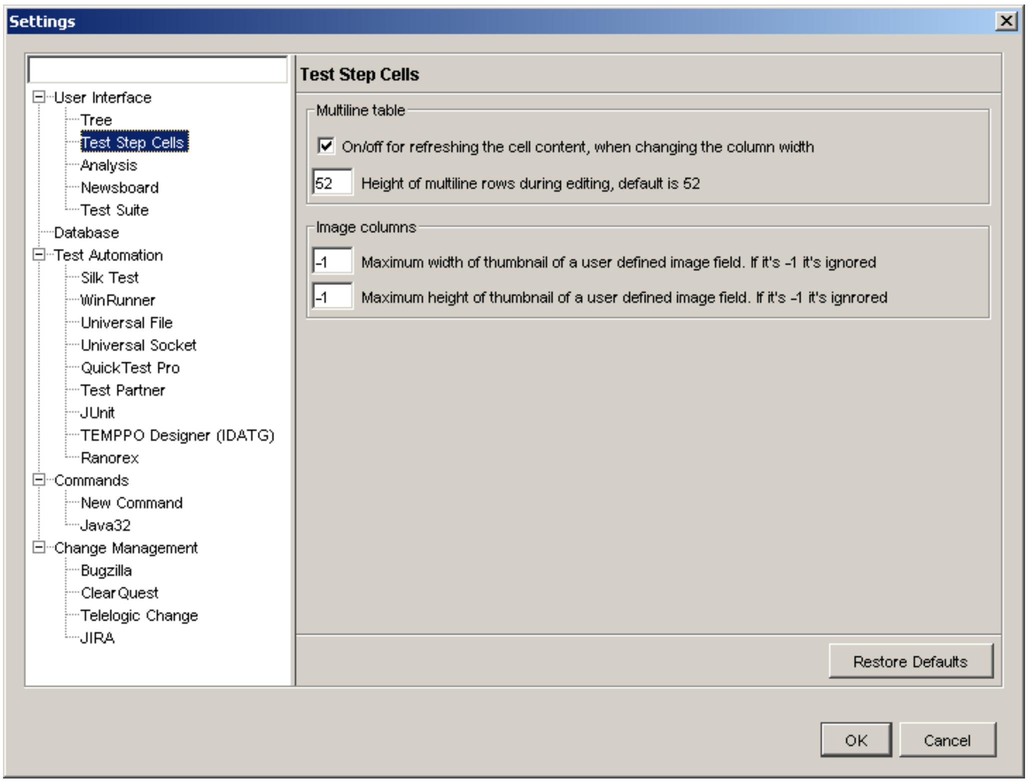

TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the requirem
3.1. Scope of delivery
3.2. Sequences (scenarios) of in
3.3. User goals
3.4. Required functions of the pr
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file o
3.4.1.1.2. New (file o
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case
3.4.4.1.1.2. Quicl
3.4.4.1.2. Find down
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of the pr
3.5.1. User interfaces
New (file opened, no changes) - tc536
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
Nr. Instruction Action image Expected
1 File.New.pick() Editor Pane cleared
Unbenannt - Editor
Datei Bearbeiten Format Ansicht ?
Neu Strg+N
Öffnen... Strg+O
Speichern Strg+S
Speichern unter...
Seite einrichten...
Drucken... Strg+P
Beenden
2 Untitled - Notepad in title bar
Upload
Delete Delete
Paste Ctrl+V
Copy
Paste image
Reference R
P+
Lock Apply Discard

Figure 53 - Upload new picture

- **Upload**: If you activate the menu **Upload** a file open dialog opens where you can select a file (jpg), which is uploaded to the database.
- **Delete**: If you activate the menu Delete, the image is deleted.
- **Paste**: If you activate the menu **Paste**, the screen shot which is currently in the clipboard is “pasted” to the cell and also uploaded to the database. The menu is disabled, if the clipboard is empty.
- **Copy**: If you activate the menu Copy, the current selected image is copied to the clipboard.
- **Reference**: If you activate the menu **Reference**, you can create a reference to an image, which is already uploaded in the database. This function is used, if you have to paste/upload the same image several times.



Test Step Cells

Multiline table

☑ On/off for refreshing the cell content, when changing the column width

52 Height of multiline rows during editing, default is 52

Image columns

-1 Maximum width of thumbnail of a user defined image field. If it's -1 it's ignored

-1 Maximum height of thumbnail of a user defined image field. If it's -1 it's ignrored

Figure 54 - TEMPPO Test Manager settings

For very large images a viewer is integrated, which can be started by double clicking the image: See Figure 55.

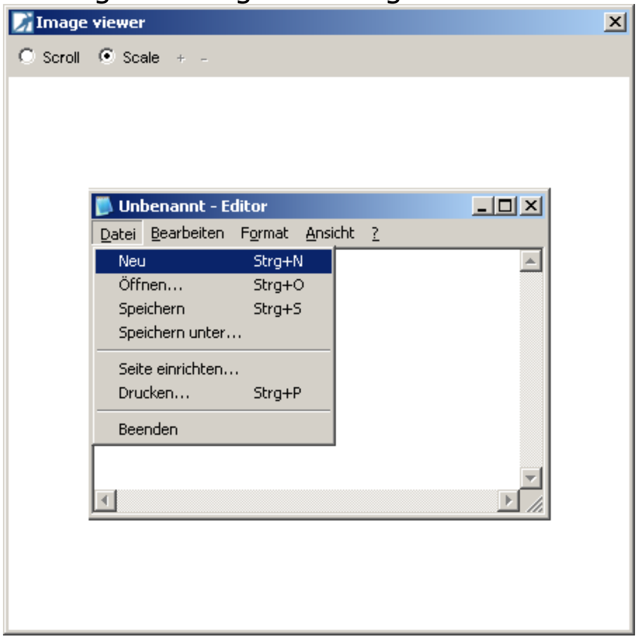


Figure 55 - External viewer

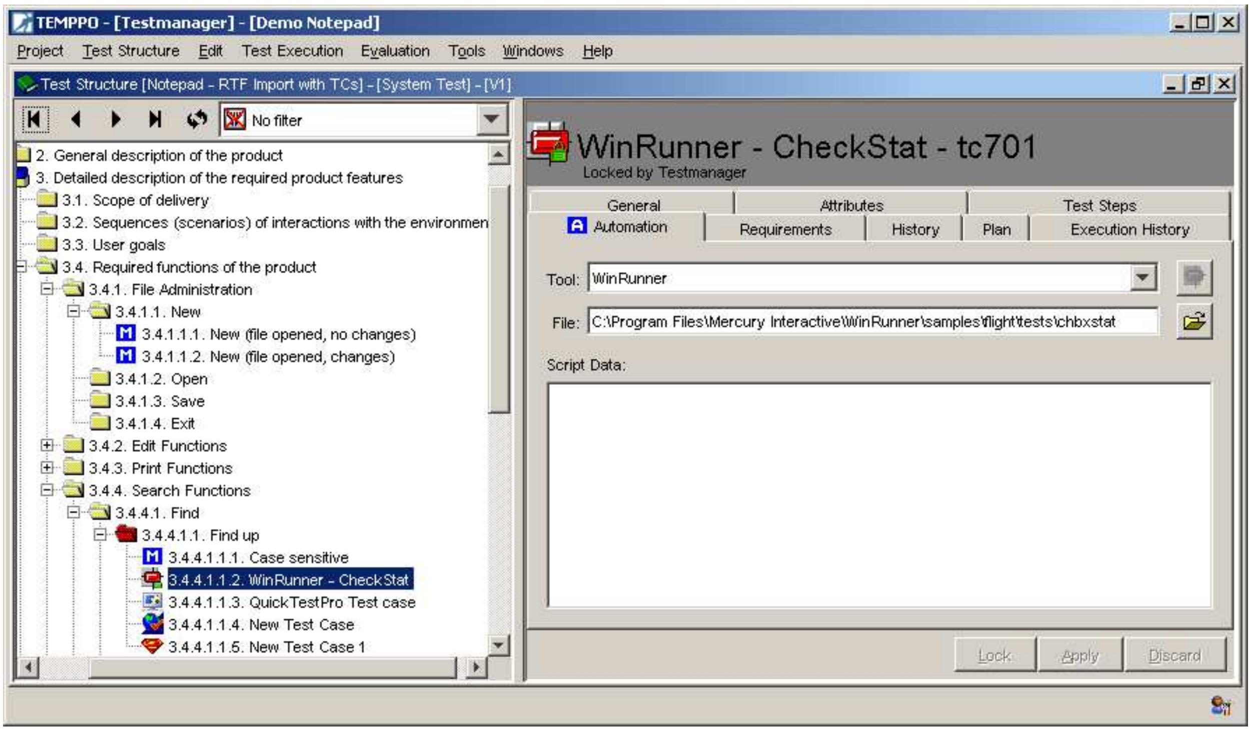
3.1.3.1.7 Automated test cases

TEMPPO can also be used to administrate automated test cases. Here test steps are stored as a test script on the files system. On one hand TEMPPO supports the tools **WinRunner** and **QuickTest Professional** of HP (http://www.hp.com), **SilkTest** of Microfocus and **TestPartner** of Microfocus (http://www.microfocus.com), **Ranorex** (http://www.ranorex.com) and on the other hand there is a **Universal Automation Interface (UAI)**.

With the context menu item **Test automated test case** (or **Edit > Test automated test case**) the test case can be started on trial, i.e. the result of test case isn't saved. The result is shown in a message box.

3.1.3.1.7.1 WinRunner, QuickTest Professional, SilkTest, TestPartner

For specifying an automated test case it is necessary to enter the tool (WinRunner, QuickTest Professional, SilkTest, TestPartner), the file name of the script plus path (a file selection dialog is opened). The text field **Script Data** is used for passing parameters to the automation script. Currently, only SilkTest uses this feature.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No filter
2. General description of the product
3. Detailed description of the required product features
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions with the environmen
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no changes)
3.4.1.1.2. New (file opened, changes)
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case sensitive
3.4.4.1.1.2. WinRunner - CheckStat
3.4.4.1.1.3. QuickTestPro Test case
3.4.4.1.1.4. New Test Case
3.4.4.1.1.5. New Test Case 1
WinRunner - CheckStat - tc701
Locked by Testmanager
General
Attributes
Test Steps
Automation
Requirements
History
Plan
Execution History
Lock
Apply
Discard

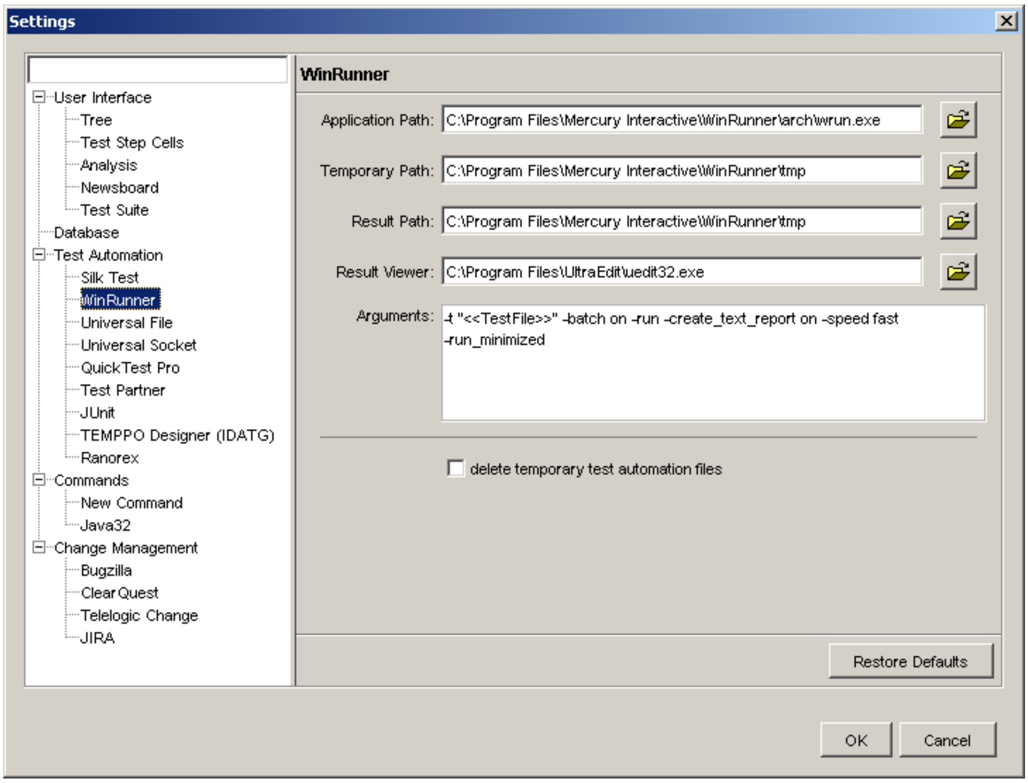
Tool: WinRunner

File: C:\Program Files\Mercury Interactive\WinRunner\samples\flight\tests\chbxstat

Script Data:

Figure 56 - Automated (WinRunner) test case

The paths to the WinRunner, QuickTest Professional, SilkTest, TestPartner executables are specified in TEMPPO settings(see Figure 57).



WinRunner

Application Path: C:\Program Files\Mercury Interactive\WinRunner\arch\wrun.exe

Temporary Path: C:\Program Files\Mercury Interactive\WinRunner\tmp

Result Path: C:\Program Files\Mercury Interactive\WinRunner\tmp

Result Viewer: C:\Program Files\UltraEdit\uedit32.exe

Arguments: -t "<<TestFile>>" -batch on -run -create_text_report on -speed fast -run_minimized

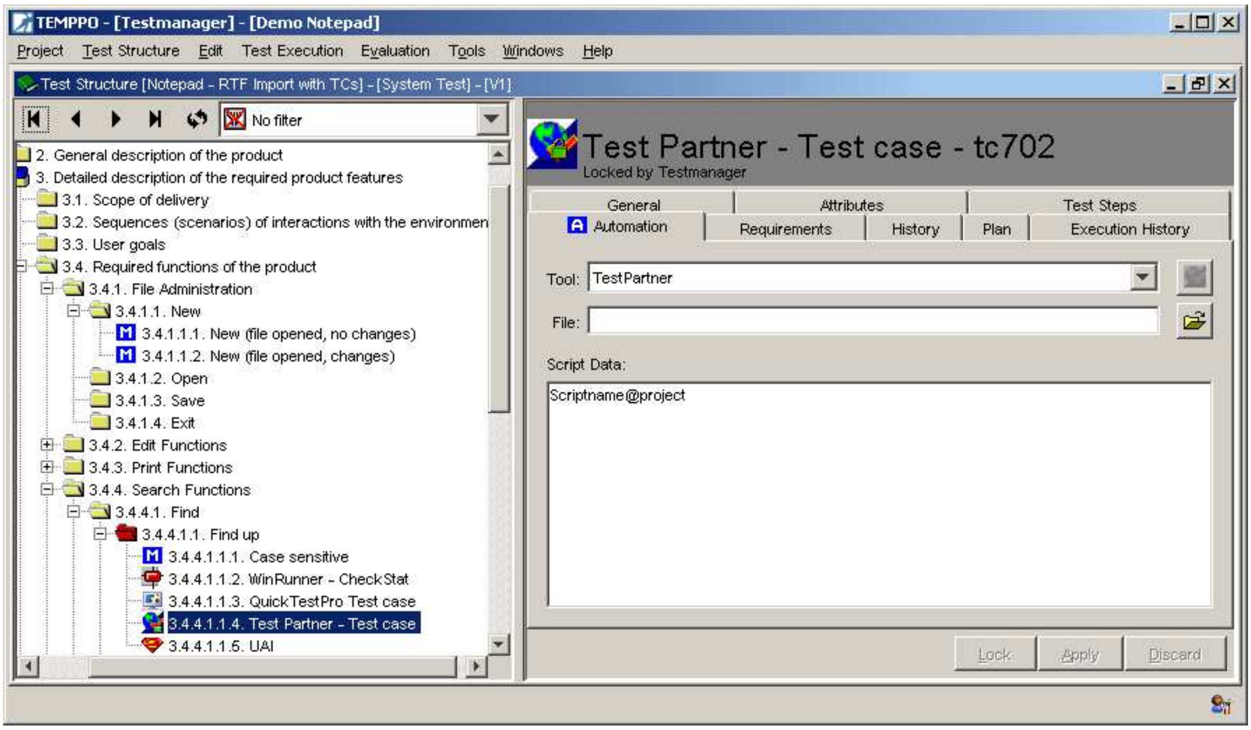
☐ delete temporary test automation files

Figure 57 - TEMPPO - Test Automation Settings

Set WinRunner mode to “Verify”. The default mode of WinRunner is "Debug", which means that no log file is written. TEMPPO needs a log file to parse and extract the results. If no log file is written, TEMPPO shows an error message:

C:\Temp\TEMPPO18072005112300\db\crvx.asc (Das System kann den angegebenen Pfad nicht finden)

TestPartner
For specifying an automated test case it is necessary to select the `tool` TestPartner. The text field `Script Data` is used for entering the script name. The convention is <script name>@<Testpartner project name>.
The text field `file` is obsolete.



2. General description of the product
3. Detailed description of the required product features
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions with the environmen
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no changes)
3.4.1.1.2. New (file opened, changes)
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case sensitive
3.4.4.1.1.2. WinRunner - CheckStat
3.4.4.1.1.3. QuickTestPro Test case
3.4.4.1.1.4. Test Partner - Test case
3.4.4.1.1.5. UAI

Tool: TestPartner
File:
Script Data:
Scriptname@project

Figure 58 - Create TestPartner test case

3.1.3.1.7.2 Ranorex

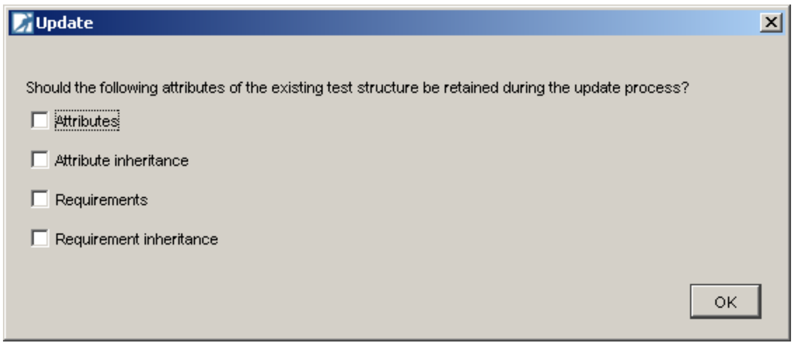
For importing test cases and folders from Ranorex you have to select test structure root, activate the menu item **Import structure -> from Ranorex Document...** and a open file dialog is displayed where you can select a Ranorex project file (.**csproj**). After pressing the button **open** the file is imported and test cases and test packages are generated. The following table shows the mapping:

Ranorex	TEMPPO
Test suite	Test structure or part of it
Test case	Test Case
Folder	Test package
Record	None

Table 1 - Ranorex - TEMPPO mapping

After importing a Ranorex test suite, you can add, change, or delete test cases in the Ranorex studio. When finished you can update the existing TEMPPO test cases and test packages by activating the menu item **Update structure -> from Ranorex Document...,** selecting the corresponding .**csproj** file and pressing the button **Open**. If the update process is started, in Figure 59 you can select the properties (attributes, attribute inheritance, requirements, requirement inheritance) of test packages and test cases which should be retained after the update.

You can import test cases, add meta information in TEMPPO, change them in Ranorex, update them in TEMPPO and keep the meta information.



Should the following attributes of the existing test structure be retained during the update process?

□ Attributes

□ Attribute inheritance

□ Requirements

□ Requirement inheritance

Figure 59 - Retaining existing test case / package attributes

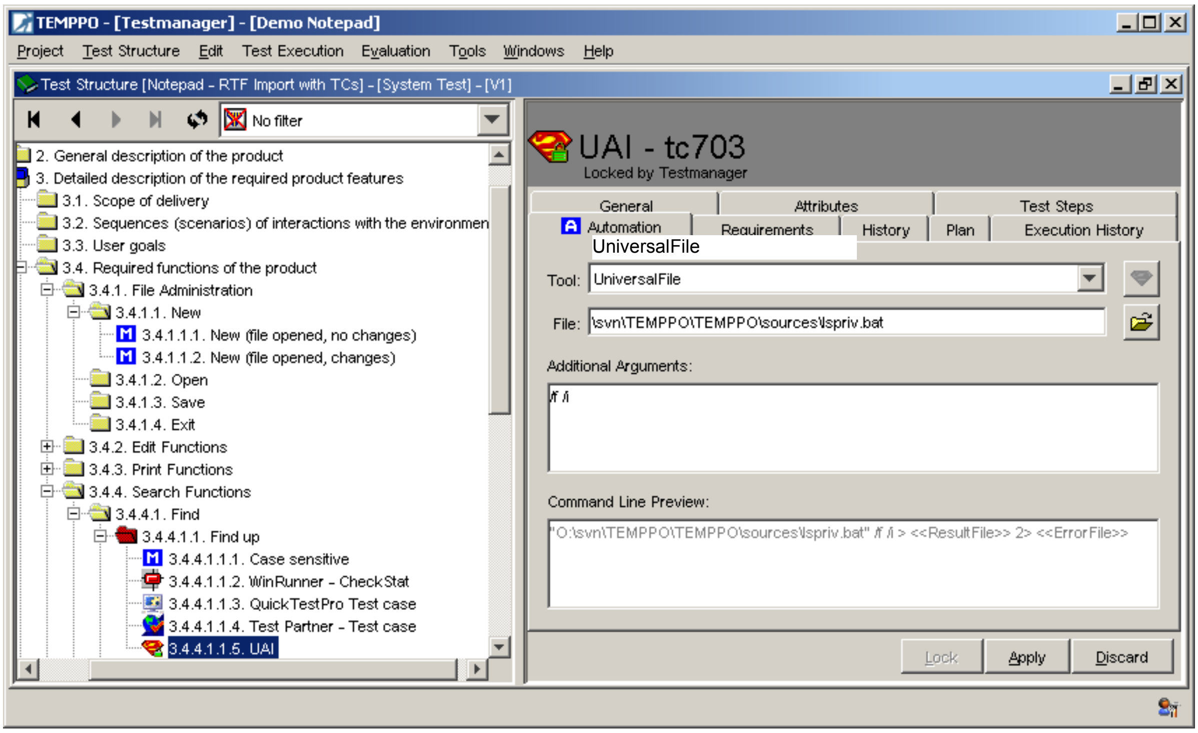
3.1.3.1.7.3 Universal Automation Interface (UAI) - File based

The purpose of this interface is to provide a possibility to manage test cases of proprietary automation tools in TEMPPO. By this, TEMPPO is independent from an automation tool. You can create an automated test case by specifying any executable (.exe, .bat, .pl,....) and the parameters, which are handed over to the executable for test case execution. The test case result is written to a file, TEMPPO parses this file and sets the result within the test suite appropriately.

First of all you have to set some general parameters in the TEMPPO settings.

For creating such a test case you have to select **UniversalFile** (Figure 60 - Test case settings for Universal Automation Interface) and fill in the following data. In the text field `File` you have to enter the path (without drive letter, see chapter 3.3.20.2.7.2) and the file name of the executable (or you use the File Chooser). In the **Additional Arguments** field you can specify command line parameters, which will be added to the arguments specified in the Administrator and passed to the executable during test case execution.

In the **Command Line Preview** the whole list of parameters is displayed.



TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1]
No filter

2. General description of the product
3. Detailed description of the required product features
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions with the environmen
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no changes)
3.4.1.1.2. New (file opened, changes)
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case sensitive
3.4.4.1.1.2. WinRunner - CheckStat
3.4.4.1.1.3. QuickTestPro Test case
3.4.4.1.1.4. Test Partner - Test case
3.4.4.1.1.5. UAI

UAI - tc703
Locked by Testmanager

Tool: UniversalFile

File: \svn\TEMPPO\TEMPPO\sources\lspriv.bat

Additional Arguments:

/f /i

Command Line Preview:

"O:\svn\TEMPPO\TEMPPO\sources\lspriv.bat" /f /i > <<ResultFile>> 2> <<ErrorFile>>

Figure 60 - Test case settings for Universal Automation Interface

Results of test cases:

All executables must create an output that contains at least this line:

result=(passed|failed|blocked|not implemented|on hold|not completed)

Additionally, TEMPPO can store an execution summary (stored to the text field **Result Output)**, if the following block is written by the test executable to the output:

summary_begin
....
....
....
summary_end

After execution of each test case, TEMPPO Test Manager will read the corresponding result file. If the whole file is missing or if the result line is missing (e.g. due to a crash of the test script), the result of the corresponding test case will be set to "**blocked**".

Otherwise, the result of the test cases' execution will be set to the value found in the result file. If the summary block is present, TEMPPO Test Manager will store the information found between start and end line in its data base.

Getting the output:

By default, TEMPPO Test Manager expects the result output to be produced on the stdout stream of the executable. For an executable

MyExec.exe

TEMPPO generates a result file named

yyyymmdd_hhmm_MyExec.out

and redirects stdout of *MyExec.exe* to this file. The prefix is a date-time stamp that denotes the time when the test executable was started. This file is stored in the directory specified as **Result Path** in the TEMPPO Test Manager settings – Test Automation.

Similar to result files, TEMPPO Test Manager expects error output of test executables to be printed to stderr. TEMPPO Test Manager will generate an error file

yyyymmdd_hhmm_MyExec.err

and redirect stderr of *MyExec.exe* to this file. The file is put to the directory specified as `Error Path` in the TEMPPO settings – Test Automation.

If the script does not write to stdout or stderr, the user can use the built-in variables <<ResultFile>> and <<ErrorFile>> in the arguments settings. The variables will be replaced at execution time by the paths specified in the TEMPPO Test Manager **Settings** plus the file names generated by TEMPPO Test Manager (see above).

ⓘ **If the executable expects the file name for the result or error output as command argument, they have to be specified by using <<ResultFile>> or <<ErrorFile>>, otherwise TEMPPO cannot read them.**

Example:

All scripts should take the command line parameters "/o myResults.res /e myErrors.err". Additionally, each script should take some individual parameters. Assume*, test_case.bat* corresponds to a TEMPPO test case and should be started with individual parameters "/f /i", like this:

```
test_case.bat /f /i /o myResults.res /e myErrors.err
```

In this case, the user specifies the general settings for all test cases in the TEMPPO Test Manager **Settings**:

```
Arguments:        /o <<ResultFile>> /e <<ErrorFile>>
```

and the individual settings in the TEMPPO test case automation tab:

```
File:                   \temppo\test_case.bat
Additional Arguments:   /f /i
```

If we assume a drive letter setting "U:\", the preview window shows a full command line like this:

```
U:\temppo\test_case.bat /f /i /o <<ResultFile>> /e <<ErrorFile>>
```

where <<ResultFile>> and <<ErrorFile>> will be replaced with generated file names (including the paths specified in the Administrator) at execution time.

For example <<ResultFile>> will be replaced with:

```
C:\Documents and Settings\atw1t5q2\Result\20021202_1723_test_case.out
```

3.1.3.2 Reworking the Test Structure

Possibly you will have to rework your test structure after a review or due to changes of the software under test.

Editing of the test structure is supported by the Cut / Copy / Paste functionality

Edit > Cut / Copy / Paste

which even allows you to copy or move around complete test structure sub-trees. By using

Edit > Order > Move Up / Move Down

you can change the order of test packages or test cases as well.

ⓘ **Edit and evaluation functionalities are also available over the context menu.**

3.1.4 Test Case Execution

Test execution is an issue, which must to some extent be connected with configuration management (CM) concepts like checking in a version and making it reproducible for later reference. If you are unfamiliar with CM terminology, please refer to chapter 3.3.10.

In an ideal world you would first make your planned test cases ready for execution, somehow conserve their state by means of CM functionality, execute them on your test object and record results. In reality, however, it often happens during test execution that you encounter a faulty test case. Such faults are mostly caused by undocumented changes in requirements or they slipped through quality control after test case implementation. Naturally you'll want to correct a faulty test case before executing it - and your test management tool should support this action. A different aspect is that other test cases which were already executed should certainly not be editable any more. Otherwise you could run into the problem of accidentally changing or even deleting a test step that already has a recorded result.

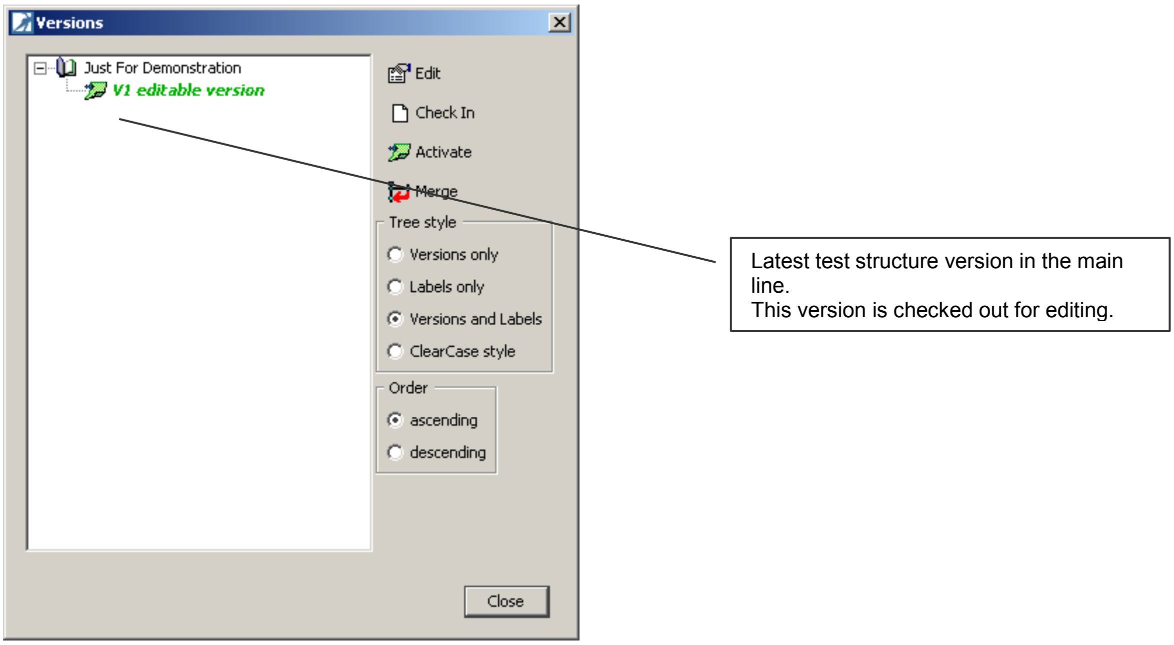
TEMPPO Test Manager supports these seemingly contradictory needs with its versioning mechanism, which is implicitly used for test execution. When you create a test suite, TEMPPO Test Manager automatically checks in your current test structure. This means that its state (test packages, test cases, test steps) is conserved: no further editing is possible within this version and it can be restored any time in future. To give you the flexibility of still being able to correct faulty test cases during test execution, another action is performed automatically at test suite creation. A branch containing a new editable version of the test structure is created (you are prompted for a descriptive name), and your new test suite is connected to it. TEMPPO allows you to edit test cases in this branch as long as no result has been recorded for any of their test steps. From then on, no further editing is possible and thus your test results cannot get lost.

Of course you can still edit test cases in a different version of your test structure, typically in the latest version on the main line. In your next test execution, again a branch off the main line will be created for you, and the corrections you applied there will be visible in the new test suite.

The complex matters of test execution are explained in greater detail in the following chapters.

3.1.4.1 Starting Test Execution

Before the first test execution, there exists usually just one version of the test structure, version "1" on the main line. In TEMPPO, every last version gets the label "latest", signaling that the test structure in this version is checked out for editing. The version tree can be displayed any time by selecting **Test Structure > Versions....** Although you never get directly in touch with the version tree during the course of test execution, references to it will be made frequently in the following paragraphs. This is necessary in order to illustrate as clearly as possible what is going on "behind the scenes" as you proceed.



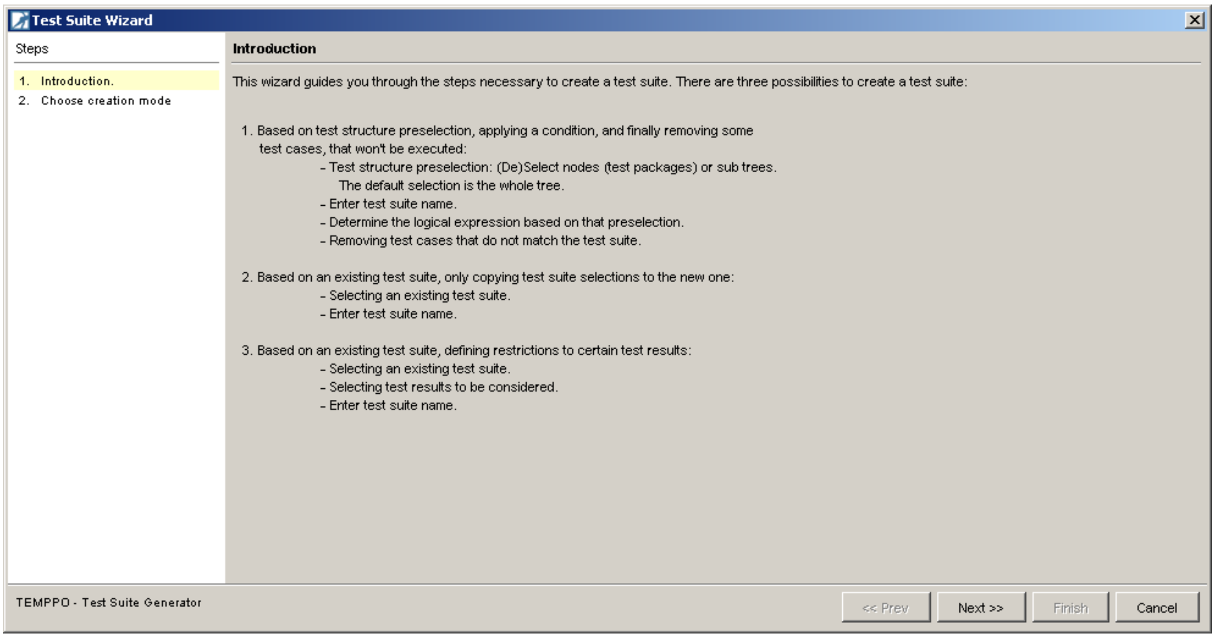
Latest test structure version in the main line.
This version is checked out for editing.

Figure 61 - Initial version tree before any test execution

Usually, when you think of starting a new test execution, you don't want all of your test cases to take part in it. Some of them may not be fully worked out, others may test features that are not yet available in the current version of your test object. For the purpose of selecting a proper subset of your test cases, TEMPPO provides the concept of test suites. By means of a logical expression, you can filter out the desired test cases.

A new Test Suite is created by activating **Test Execution > New**. This will cause the wizard shown in Figure 62 to be displayed. There are 4 possibilities to create a test suite:

- **New:** Based on test structure pre-selection, applying a condition, and finally removing some test cases, that won't be executed.
- **Based on a predecessor:** new test suite is chained with an existing test suite (chain)
- **Copy**: Based on an existing test suite, only copying test suite selections to the new one.
- **Special**: Based on an existing test suite, defining restrictions on certain test results.



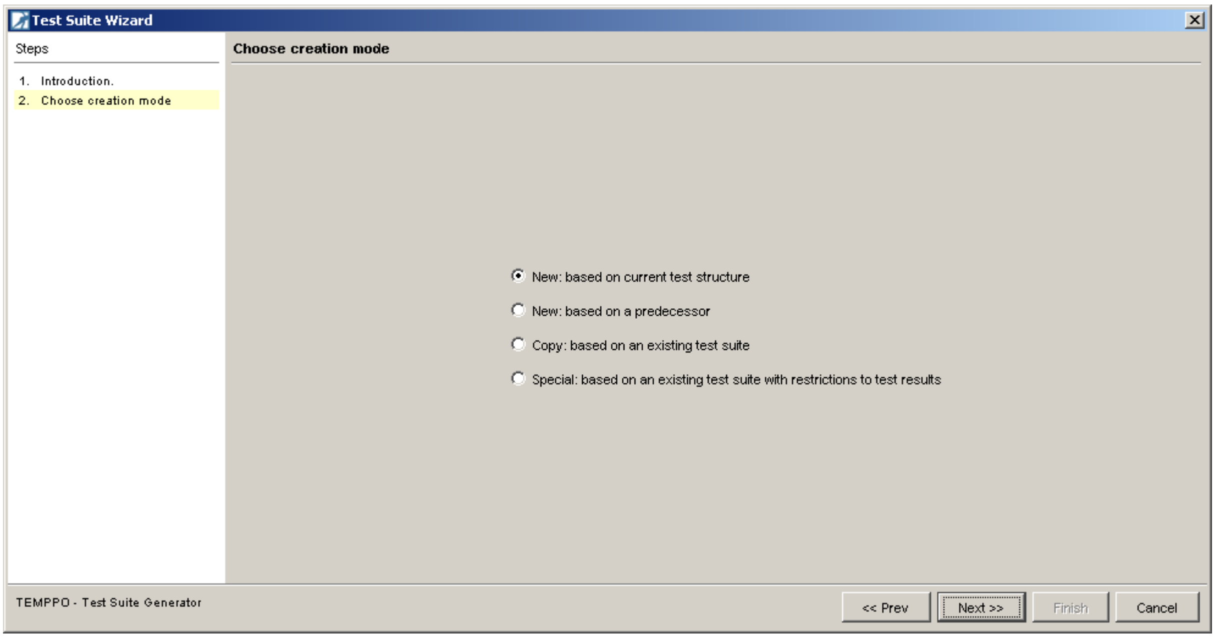
Introduction

This wizard guides you through the steps necessary to create a test suite. There are three possibilities to create a test suite:

1. Based on test structure preselection, applying a condition, and finally removing some test cases, that won't be executed:
   - Test structure preselection: (De)Select nodes (test packages) or sub trees. The default selection is the whole tree.
   - Enter test suite name.
   - Determine the logical expression based on that preselection.
   - Removing test cases that do not match the test suite.
2. Based on an existing test suite, only copying test suite selections to the new one:
   - Selecting an existing test suite.
   - Enter test suite name.
3. Based on an existing test suite, defining restrictions to certain test results:
   - Selecting an existing test suite.
   - Selecting test results to be considered.
   - Enter test suite name.

Figure 62 - Test Suite wizard: Introduction

When pressing **Next** Figure 63 is shown, where you can select one mode.



Test Suite Wizard

Steps

1. Introduction.
2. Choose creation mode

Choose creation mode

● New: based on current test structure

○ New: based on a predecessor

○ Copy: based on an existing test suite

○ Special: based on an existing test suite with restrictions to test results

TEMPPO - Test Suite Generator

Figure 63 - Creation mode selection

3.1.4.1.1 New: Based on current test structure

This mode is the usual way to create a test suite. The following steps are necessary:

- **Test structure pre-selection**
- **Test suite name**
- **Logical expression**
- **Remove test cases manually**

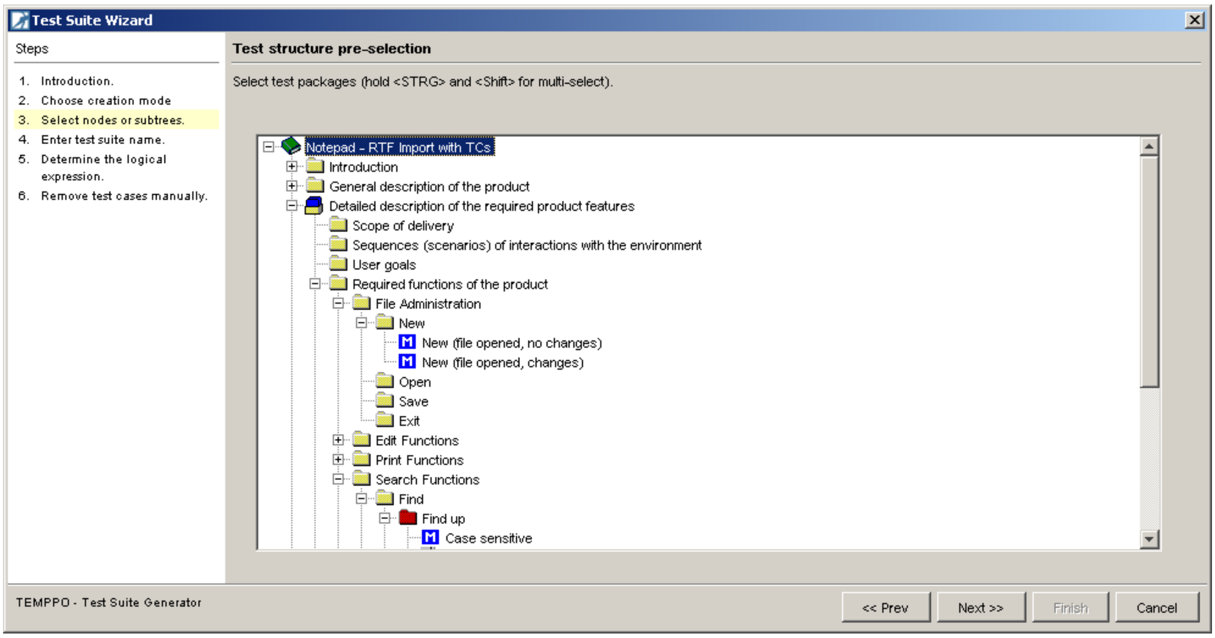

Test Suite Wizard
Steps
1. Introduction.
2. Choose creation mode
3. Select nodes or subtrees.
4. Enter test suite name.
5. Determine the logical expression.
6. Remove test cases manually.
Test structure pre-selection
Select test packages (hold <STRG> and <Shift> for multi-select).
Notepad - RTF Import with TCs
Introduction
General description of the product
Detailed description of the required product features
Scope of delivery
Sequences (scenarios) of interactions with the environment
User goals
Required functions of the product
File Administration
New
New (file opened, no changes)
New (file opened, changes)
Open
Save
Exit
Edit Functions
Print Functions
Search Functions
Find
Find up
Case sensitive
TEMPPO - Test Suite Generator
<< Prev
Next >>
Finish
Cancel

Figure 64 – New: Test structure pre-selection (whole test structure)

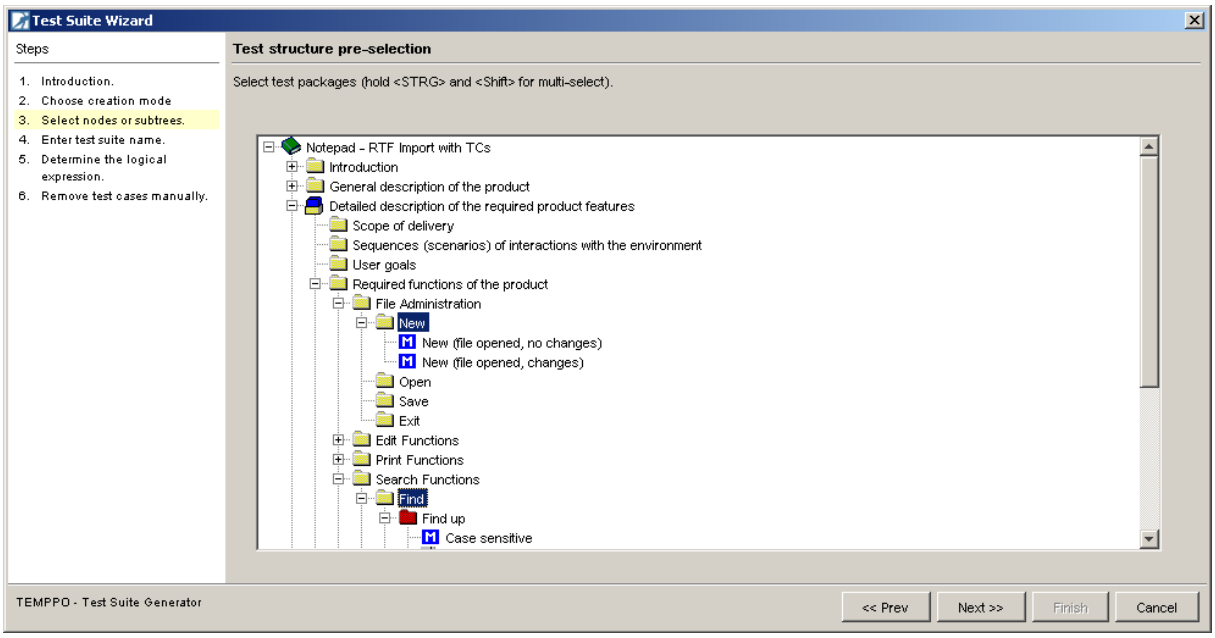
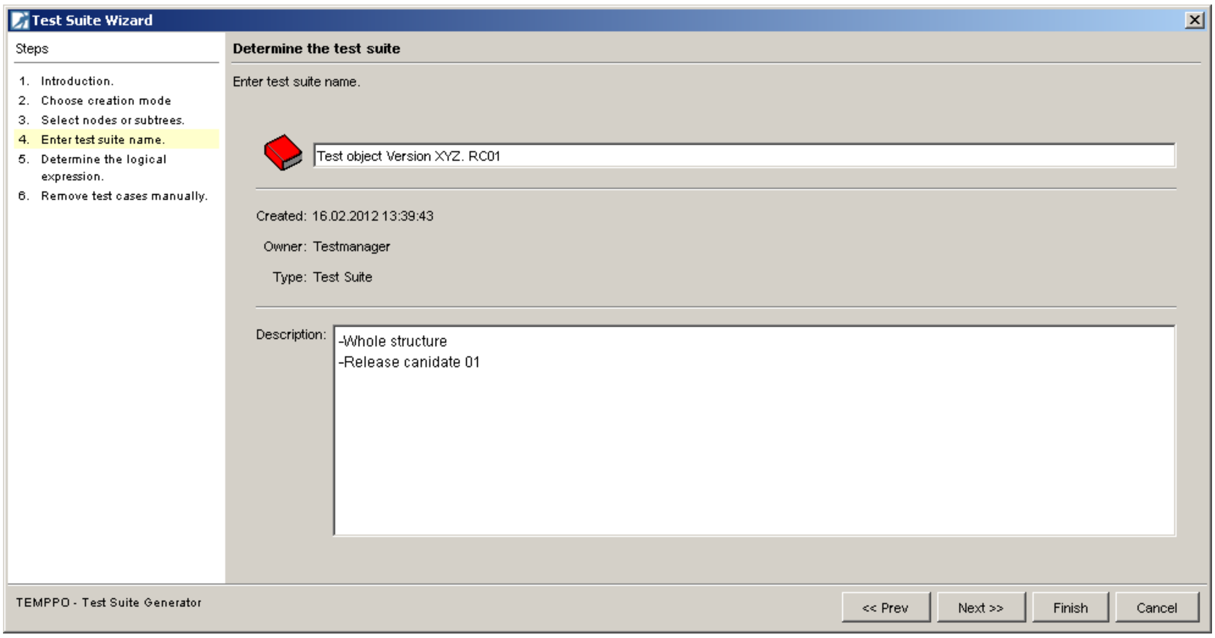

Test Suite Wizard
Steps
1. Introduction.
2. Choose creation mode
3. Select nodes or subtrees.
4. Enter test suite name.
5. Determine the logical expression.
6. Remove test cases manually.
Test structure pre-selection
Select test packages (hold <STRG> and <Shift> for multi-select).
Notepad - RTF Import with TCs
Introduction
General description of the product
Detailed description of the required product features
Scope of delivery
Sequences (scenarios) of interactions with the environment
User goals
Required functions of the product
File Administration
New
New (file opened, no changes)
New (file opened, changes)
Open
Save
Exit
Edit Functions
Print Functions
Search Functions
Find
Find up
Case sensitive
TEMPPO - Test Suite Generator
<< Prev
Next >>
Finish
Cancel

Figure 65 - New: Test structure pre selection (test packages)

Figure 64 shows the first step for creating a test suite. It shows the whole test structure pre-selected. If you want to consider a whole test structure with all the test packages and test cases for test execution, you press simply **Next**.

Advanced users can limit the test structure by selecting test package(s) that should be in the test suite. For selecting the test packages use <CTRL> or <Shift> keys.



1. Introduction.
2. Choose creation mode
3. Select nodes or subtrees.
4. Enter test suite name.
5. Determine the logical expression.
6. Remove test cases manually.

Determine the test suite

Enter test suite name.

Test object Version XYZ. RC01

Created: 16.02.2012 13:39:43
Owner: Testmanager
Type: Test Suite

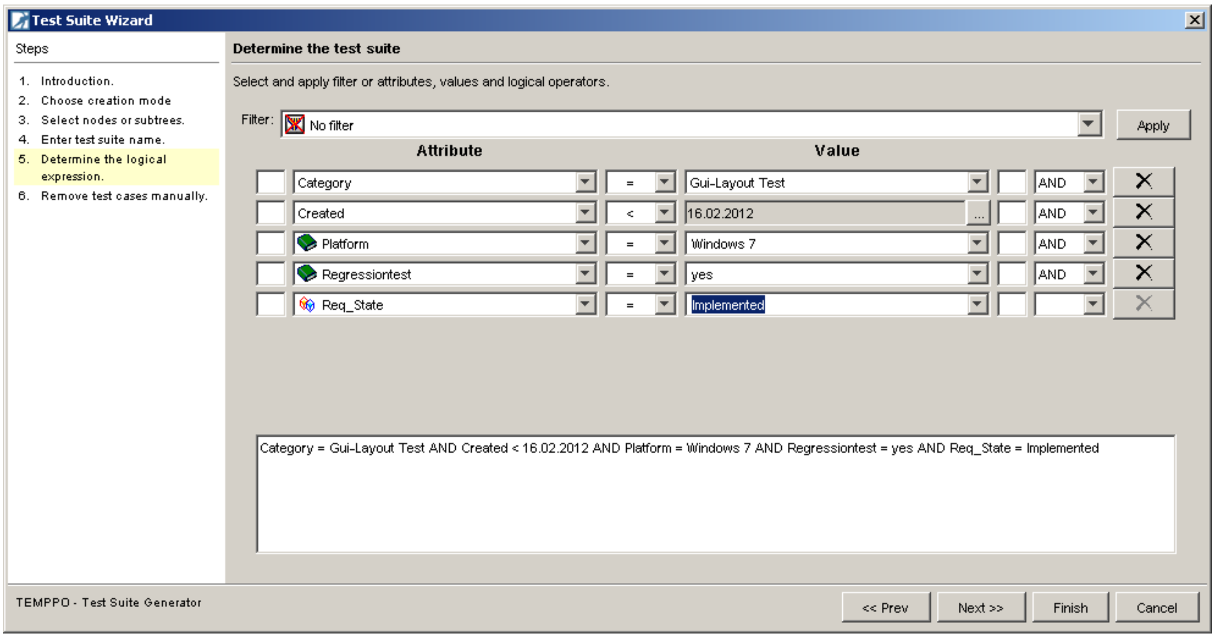
Description:
-Whole structure
-Release canidate 01

Figure 66 - New: Name and comment

In the 2nd step you have to enter **name** together with a **description** for the test suite (Figure 66). When pressing **Next** you have to determine the logical expression (Figure 67).

A logical expression can be defined, which filters out the desired test cases. It can be constructed using test package categories, test levels and test case attributes. Specifying a logical connector in the last line can extend the expression.

If you have already defined a filter for your test issue you can select it and press the button **Apply**. The filter's logical expression is applied.



1. Introduction.
2. Choose creation mode
3. Select nodes or subtrees.
4. Enter test suite name.
5. Determine the logical expression.
6. Remove test cases manually.

Determine the test suite

Select and apply filter or attributes, values and logical operators.

Filter: No filter

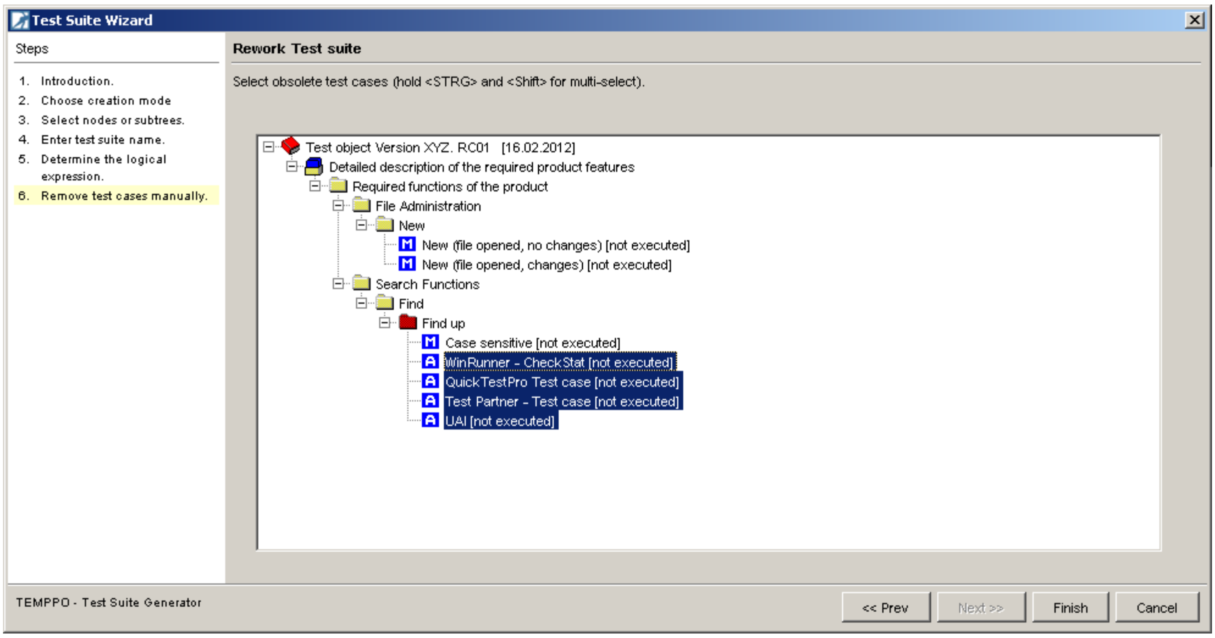
	Attribute		Value		
	Category	=	Gui-Layout Test		AND
	Created	<	16.02.2012		AND
	Platform	=	Windows 7		AND
	Regressiontest	=	yes		AND
	Req_State	=	Implemented		

Category = Gui-Layout Test AND Created < 16.02.2012 AND Platform = Windows 7 AND Regressiontest = yes AND Req_State = Implemented

Figure 67 - New: Logical expression

When pressing **Next** the user can remove test cases that should not be executed (Figure 68). The (de)selection can be realized with the <**CTRL**> and <**Shift**> keys.

For default handling (before V3.3) press **Finish** and all test cases remain in the test suite that will be displayed.



Test Suite Wizard

Steps

1. Introduction.
2. Choose creation mode
3. Select nodes or subtrees.
4. Enter test suite name.
5. Determine the logical expression.
6. Remove test cases manually.

Rework Test suite

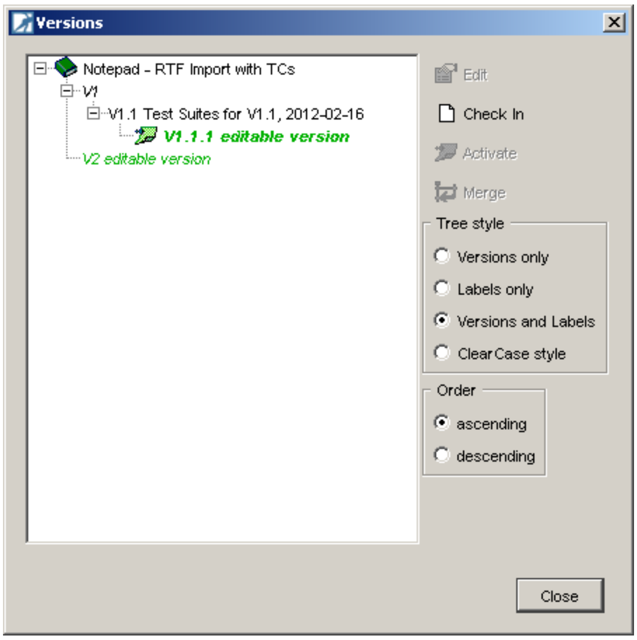
Select obsolete test cases (hold <STRG> and <Shift> for multi-select).

- Test object Version XYZ. RC01 [16.02.2012]
  - Detailed description of the required product features
    - Required functions of the product
      - File Administration
        - New
          - New (file opened, no changes) [not executed]
          - New (file opened, changes) [not executed]
      - Search Functions
        - Find
          - Find up
            - Case sensitive [not executed]
            - WinRunner - CheckStat [not executed]
            - QuickTestPro Test case [not executed]
            - Test Partner - Test case [not executed]
            - UAI [not executed]

TEMPPO - Test Suite Generator

Figure 68 - New: Remove unwanted test cases

After clicking **Finish**, the current test structure version ("V1") is checked in. If you could open the version tree browser now, you would encounter the following situation:



Versions

- Notepad - RTF Import with TCs
  - V1
    - V1.1 Test Suites for V1.1, 2012-02-16
      - V1.1.1 editable version
  - V2 editable version

Tree style
○ Versions only
○ Labels only
● Versions and Labels
○ ClearCase style

Order
● ascending
○ descending

Figure 69 - Version tree after test suite creation

ⓘ **Immediately after test suite creation, three versions with identical test structures exist: "1", "2 latest" and "1.1.1 latest". The latter two of them are editable, just the first one is checked in.**

A test suite is always just a "view" of the underlying test structure in the activated version. Only test cases are visible that fulfill the criteria defined in the logical expression. Furthermore, test packages where no test cases are selected at all are not displayed.

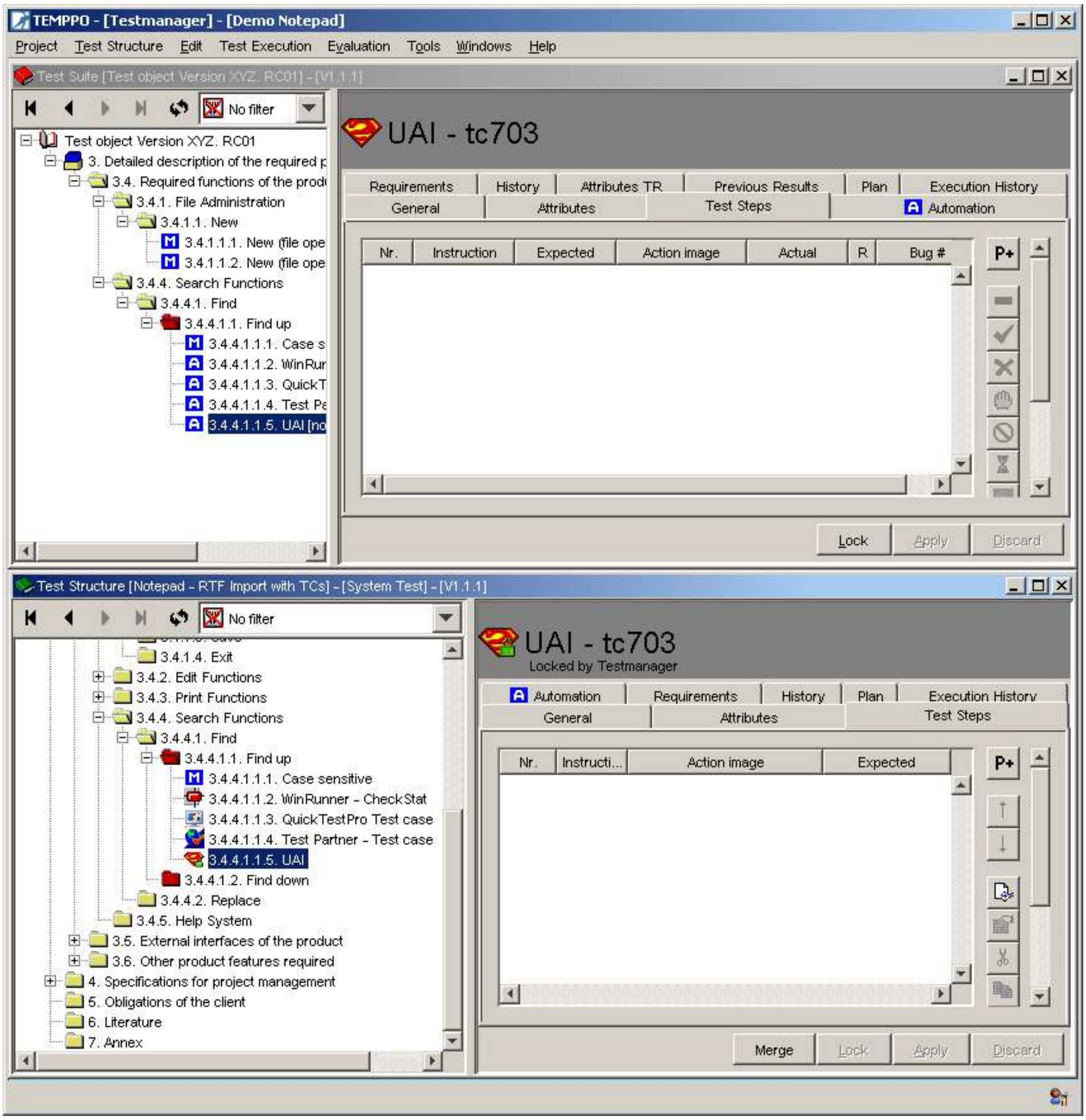
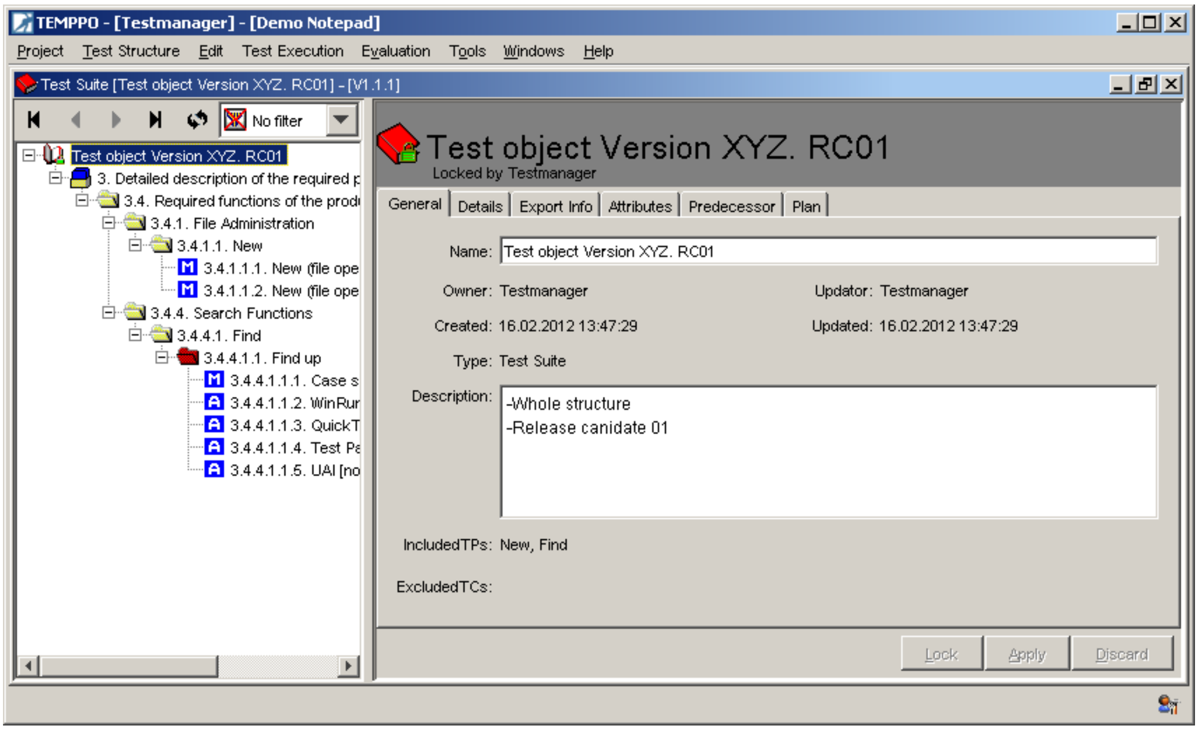


Figure 70 - Test Suite

Now that you have a test suite with all desired test cases, you are ready to start for the test execution and recording of results. Depending on the test cases, test execution is manual, automatic or mixed.

The selection information (included TPs, excluded TCs) will be displayed, if the root node of the test suite is selected (see Figure 71).


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Suite [Test object Version XYZ. RC01] - [V1.1.1]
No filter
Test object Version XYZ. RC01
3. Detailed description of the required p
3.4. Required functions of the produ
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file ope
3.4.1.1.2. New (file ope
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case s
3.4.4.1.1.2. WinRur
3.4.4.1.1.3. QuickT
3.4.4.1.1.4. Test Pa
3.4.4.1.1.5. UAI [no
Test object Version XYZ. RC01
Locked by Testmanager
General Details Export Info Attributes Predecessor Plan
Lock Apply Discard

Name: Test object Version XYZ. RC01

Owner: Testmanager

Updator: Testmanager

Created: 16.02.2012 13:47:29

Updated: 16.02.2012 13:47:29

Type: Test Suite

Description: -Whole structure
-Release canidate 01

IncludedTPs: New, Find

ExcludedTCs:

Figure 71 - Information about included TPs and excluded TCs

The tab **Details** displays the logical criterion of the test suite. It can be changed since there aren’t any results yet recorded.

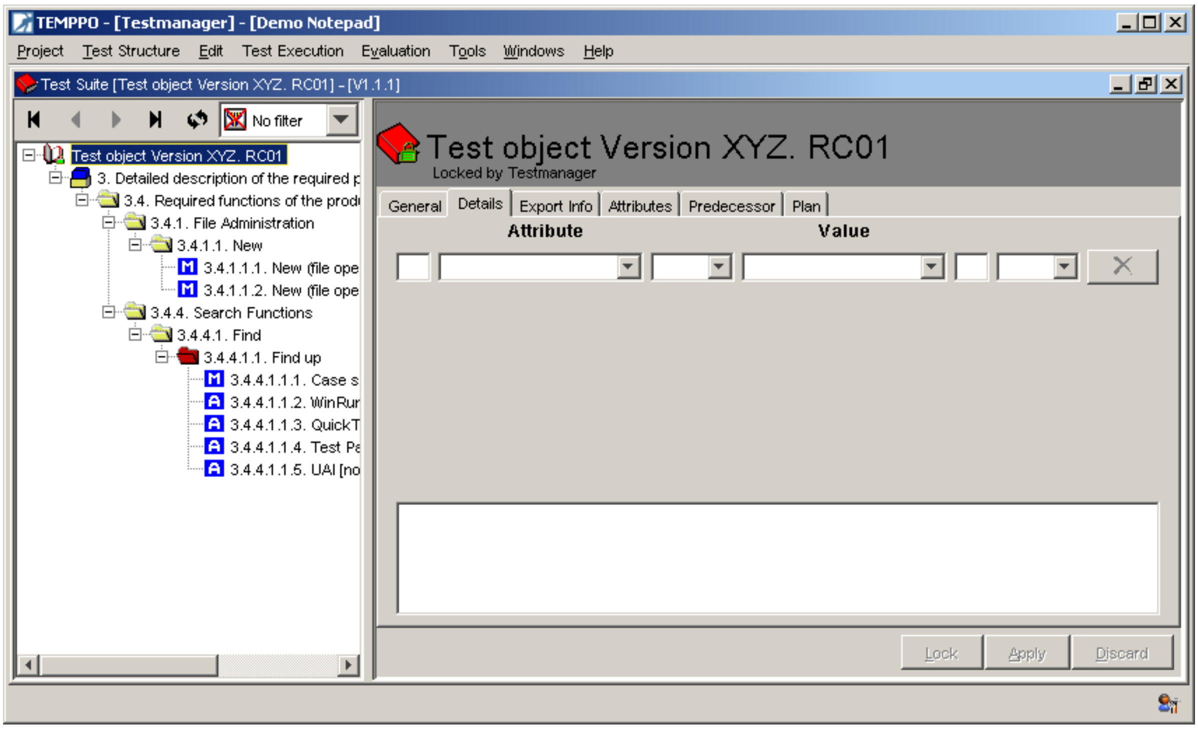
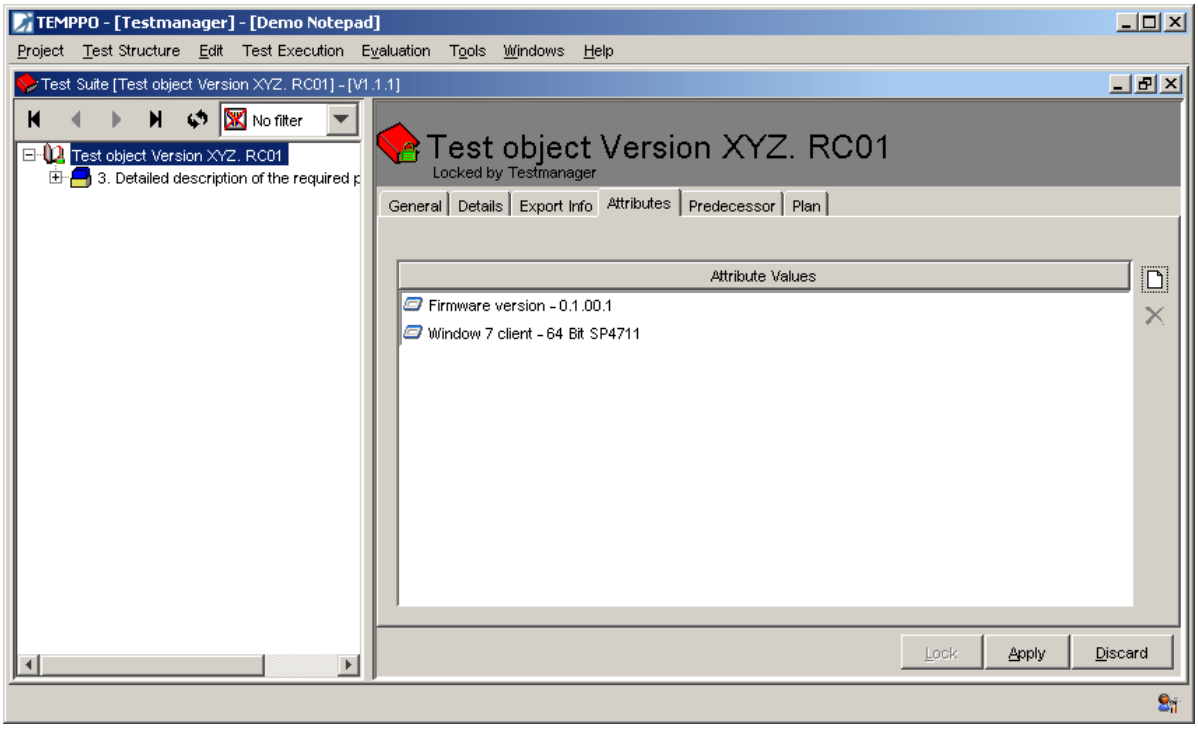

TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Suite [Test object Version XYZ. RC01] - [V1.1.1]
No filter
Test object Version XYZ. RC01
3. Detailed description of the required p
3.4. Required functions of the produ
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file ope
3.4.1.1.2. New (file ope
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case s
3.4.4.1.1.2. WinRur
3.4.4.1.1.3. QuickT
3.4.4.1.1.4. Test Pa
3.4.4.1.1.5. UAI [no
Test object Version XYZ. RC01
Locked by Testmanager
General Details Export Info Attributes Predecessor Plan
Attribute
Value
Lock Apply Discard

Figure 72 - Test suite - logical expression

In the test suite tab **Attributes**, you can set attributes, which can also be shown in the report.

Only attributes can be chosen that are indicated as test suite attributes in the Meta Data Editor (see **5.3**).


TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Suite [Test object Version XYZ. RC01] - [V1.1.1]
No filter
Test object Version XYZ. RC01
3. Detailed description of the required p
Test object Version XYZ. RC01
Locked by Testmanager
General
Details
Export Info
Attributes
Predecessor
Plan
Lock
Apply
Discard

Attribute Values
Firmware version - 0.1.00.1
Window 7 client - 64 Bit SP4711

Figure 73 - Attributes

3.1.4.1.2New: Based on a predecessor

This mode is used to create a new test suites for sprints. If your test cycle consists of 4 sprints a summarized test is necessary. With this feature you are able to “chain” test suites and to generate an overall report.

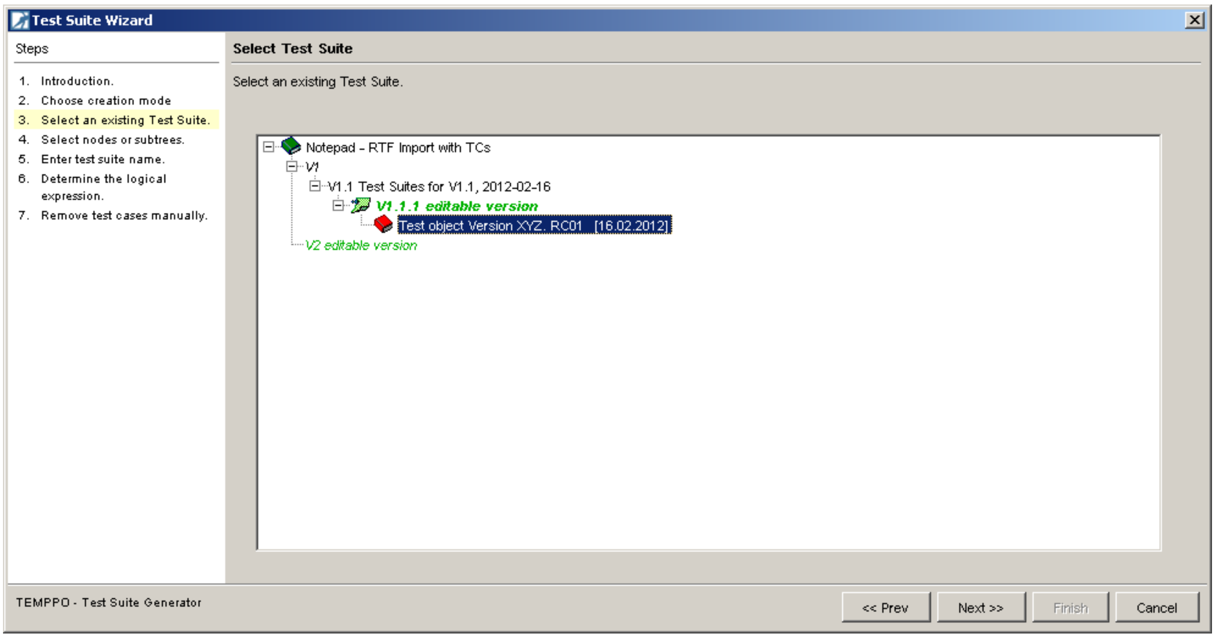
The test suite will be created on a branch of the current selected test structure version. Therefore it is possible to set a predecessor test suite and all possible settings are pre-selected.

With defining predecessors, you get information of previous test executions and previous results of each test case with bug IDs. It’s not necessary anymore to open another test suite for getting the result and bug ID from a previous test execution of a test case.

The following steps are necessary to do:

- **Select an existing test suite**
- **Test structure pre-selection**
- **Test suite name**
- **Logical expression**
- **Remove test cases manually**

In the first step the test suite tree is shown (Figure 80), where you can select an existing test suite, that will be used for pre-select all settings in the wizard.



Test Suite Wizard

Steps

1. Introduction.
2. Choose creation mode
3. Select an existing Test Suite.
4. Select nodes or subtrees.
5. Enter test suite name.
6. Determine the logical expression.
7. Remove test cases manually.

Select Test Suite

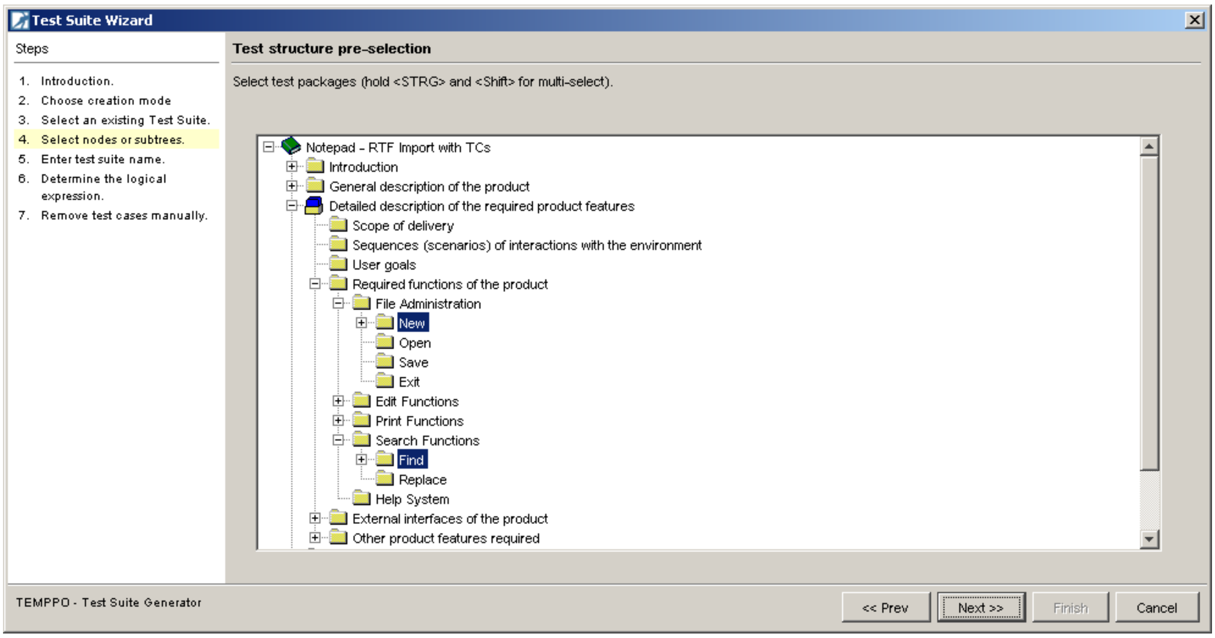
Select an existing Test Suite.

```
Notepad - RTF Import with TCs
  V1
    V1.1 Test Suites for V1.1, 2012-02-16
      V1.1.1 editable version
        Test object Version XYZ. RC01   [16.02.2012]
  V2 editable version
```

Figure 74 - Select a test suite

Figure 75 shows the first step for creating a test suite. It shows the whole test structure and the test packages are pre-selected like the predecessor.

If you want to consider a whole test structure with all the test packages and test cases for test execution, select the root node and press **Next**.



Test Suite Wizard

Steps

1. Introduction.
2. Choose creation mode
3. Select an existing Test Suite.
4. Select nodes or subtrees.
5. Enter test suite name.
6. Determine the logical expression.
7. Remove test cases manually.

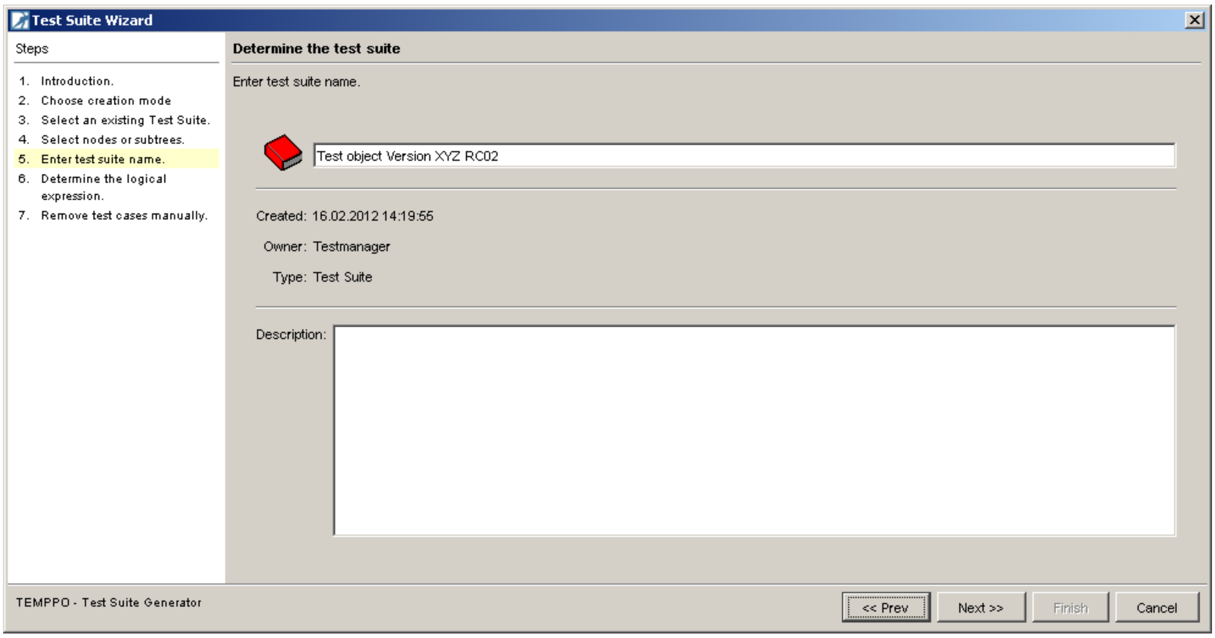
Test structure pre-selection

Select test packages (hold <STRG> and <Shift> for multi-select).

```
Notepad - RTF Import with TCs
  Introduction
  General description of the product
  Detailed description of the required product features
    Scope of delivery
    Sequences (scenarios) of interactions with the environment
    User goals
    Required functions of the product
      File Administration
        New
        Open
        Save
        Exit
      Edit Functions
      Print Functions
      Search Functions
        Find
        Replace
      Help System
    External interfaces of the product
    Other product features required
```

Figure 75 – Test structure pre-selection (test packages

In the 3rd step you have to enter **name** together with a **description** for the test suite (Figure 76). When pressing **Next** you have to determine the logical expression (Figure 77) which is pre-selected like the predecessor.



Test Suite Wizard
Steps
1. Introduction.
2. Choose creation mode
3. Select an existing Test Suite.
4. Select nodes or subtrees.
5. Enter test suite name.
6. Determine the logical expression.
7. Remove test cases manually.
Determine the test suite
TEMPPO - Test Suite Generator
<< Prev
Next >>
Finish
Cancel

Enter test suite name.

Test object Version XYZ RC02

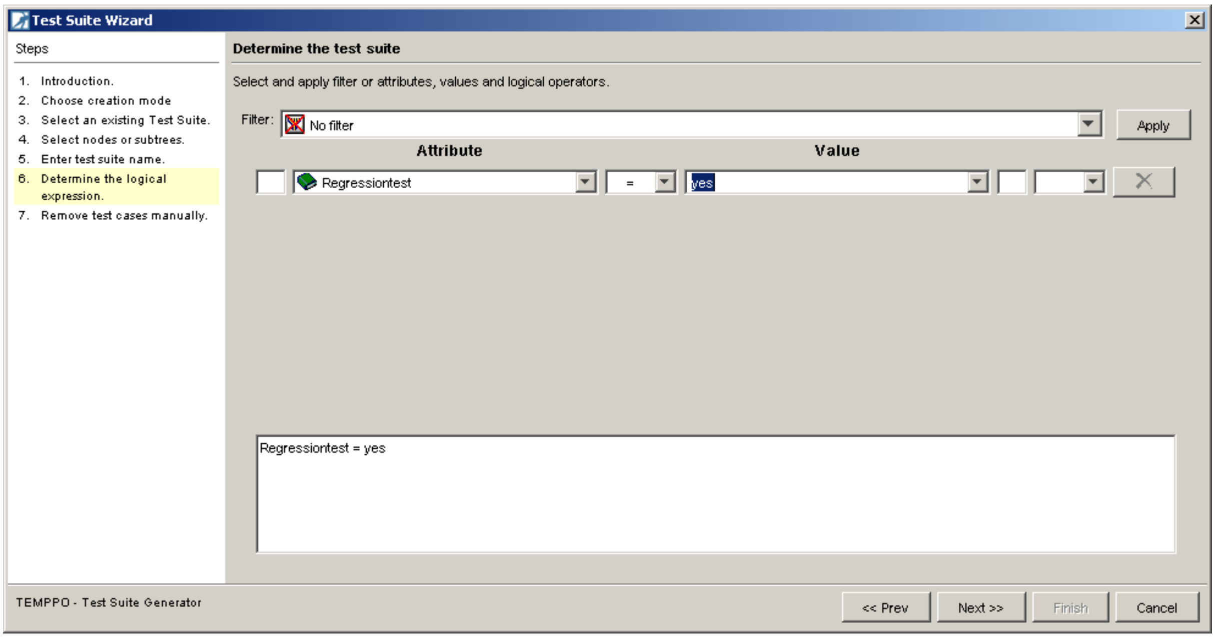
Created: 16.02.2012 14:19:55

Owner: Testmanager

Type: Test Suite

Description:

Figure 76 – Name and comment



Test Suite Wizard
Steps
1. Introduction.
2. Choose creation mode
3. Select an existing Test Suite.
4. Select nodes or subtrees.
5. Enter test suite name.
6. Determine the logical expression.
7. Remove test cases manually.
Determine the test suite
Apply
Attribute
Value
TEMPPO - Test Suite Generator
<< Prev
Next >>
Finish
Cancel

Select and apply filter or attributes, values and logical operators.

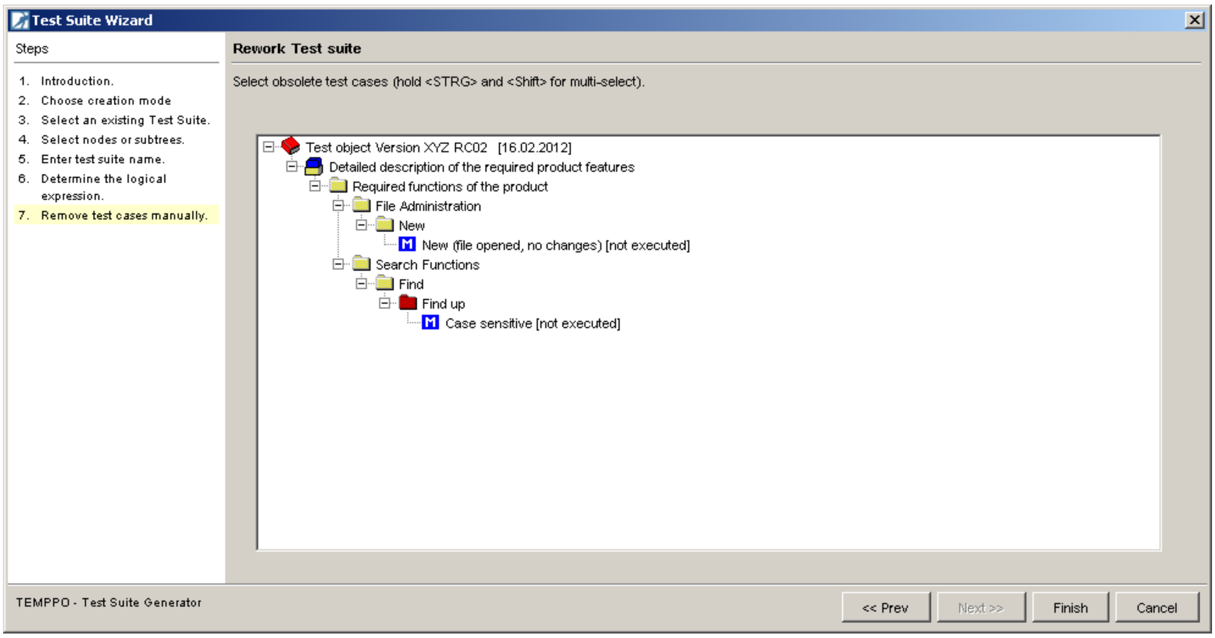
Filter: No filter

Regressiontest = yes

Regressiontest = yes

Figure 77 – Logical Expression

When pressing **Next** the user can remove test cases that should not be executed (Figure 78). The test cases are also pre-selected like the predecessor was set. The (de)selection can be realized with the <**CTRL**> and <**Shift**> keys.



Test Suite Wizard
Steps
1. Introduction.
2. Choose creation mode
3. Select an existing Test Suite.
4. Select nodes or subtrees.
5. Enter test suite name.
6. Determine the logical expression.
7. Remove test cases manually.
TEMPPO - Test Suite Generator
<< Prev
Next >>
Finish
Cancel

Rework Test suite

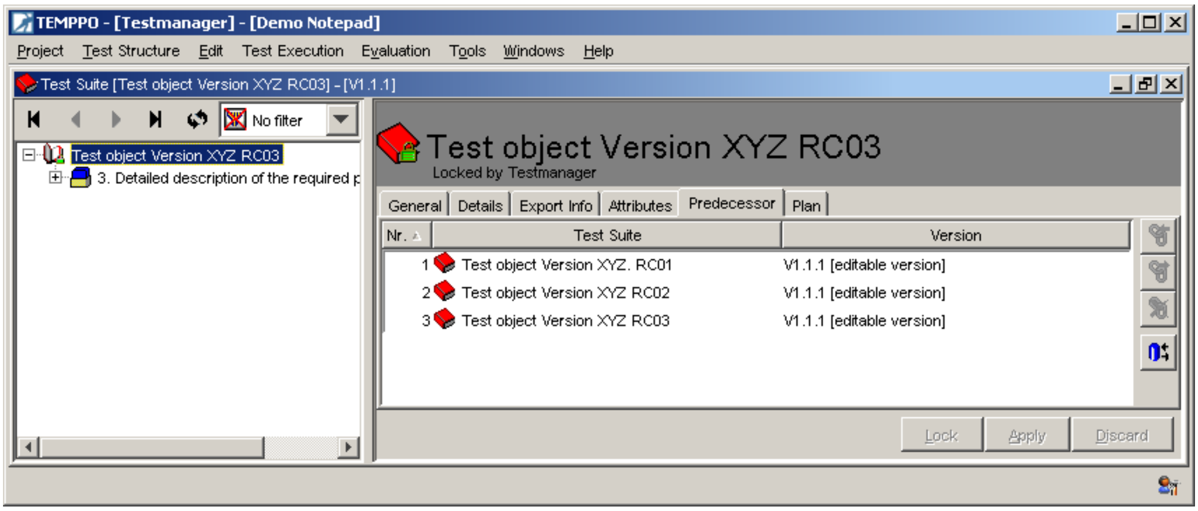
Select obsolete test cases (hold <STRG> and <Shift> for multi-select).

```
Test object Version XYZ RC02   [16.02.2012]
  Detailed description of the required product features
    Required functions of the product
      File Administration
        New
          New (file opened, no changes) [not executed]
      Search Functions
        Find
          Find up
            Case sensitive [not executed]
```

Figure 78 – Remove unwanted test cases

After clicking **Finish**, the new test suite is created based on the settings of the predecessor test suite. Now that you have a test suite with all desired test cases, you are ready to start for the test execution and recording of results.

The predecessor information will be displayed, if the root node of test suite is selected and tab Predecessor is activated.



TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Suite [Test object Version XYZ RC03] - [V1.1.1]
No filter
Test object Version XYZ RC03
3. Detailed description of the required p
Test object Version XYZ RC03
Locked by Testmanager
General
Details
Export Info
Attributes
Predecessor
Plan
Lock
Apply
Discard

Nr.	Test Suite	Version
1	Test object Version XYZ. RC01	V1.1.1 [editable version]
2	Test object Version XYZ RC02	V1.1.1 [editable version]
3	Test object Version XYZ RC03	V1.1.1 [editable version]

Figure 79 - Predecessor

The predecessors of a test suite can be changed with the buttons **Insert Into Chain, Remove Chain** and **Remove From Chain**. The predecessors are sorted by the sequence so that the last row is the current test suite. The sort can be changed with the combo box **Sort bottom-up**.

With the change of predecessors the settings of the current test suite can’t be changed in common.

Now all results of one test case are displayed in the new tab **Previous Results** (see 3.1.4.1.8). All results can be analyzed (see 3.1.5.1.1) and written to report (see Figure 153), so that all results of a test case in a sequence of test suites can be shown.

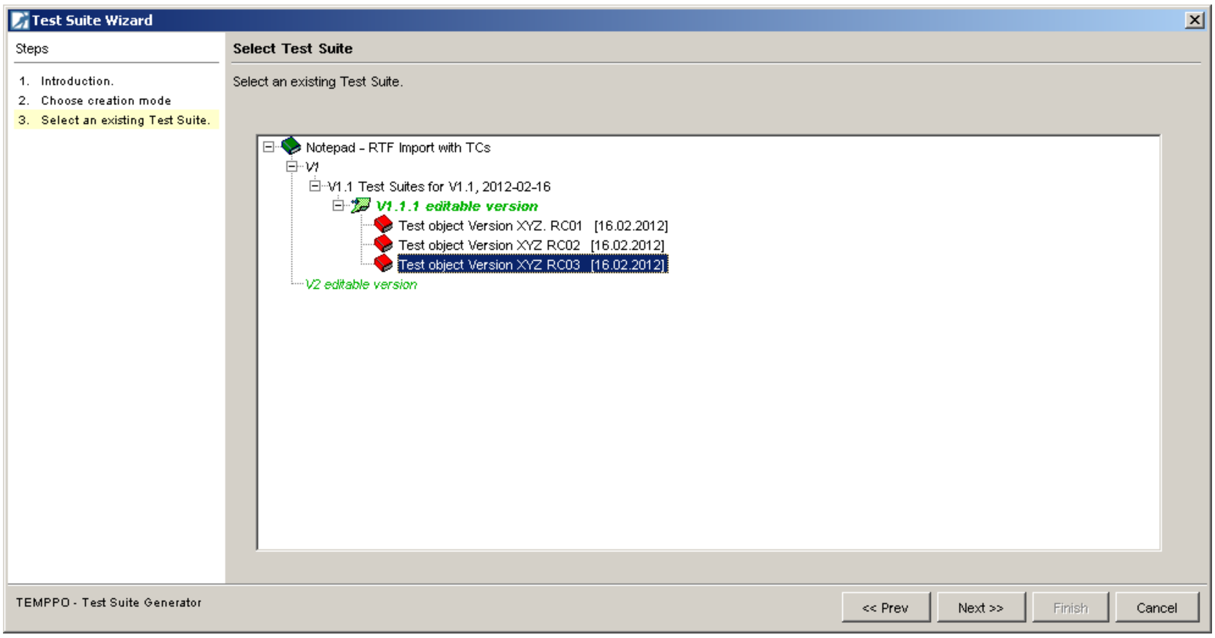
3.1.4.1.3 Copy: Based on an existing test suite

This mode is used to copy a test suite and repeat a test run with exactly the same test cases. The copied test suite will be created on a branch of the current selected test structure version. Therefore it is possible to copy test suites from prior version to later ones and vice versa.

The following steps are necessary to do:

- **Select an existing test suite**
- **Test suite name**

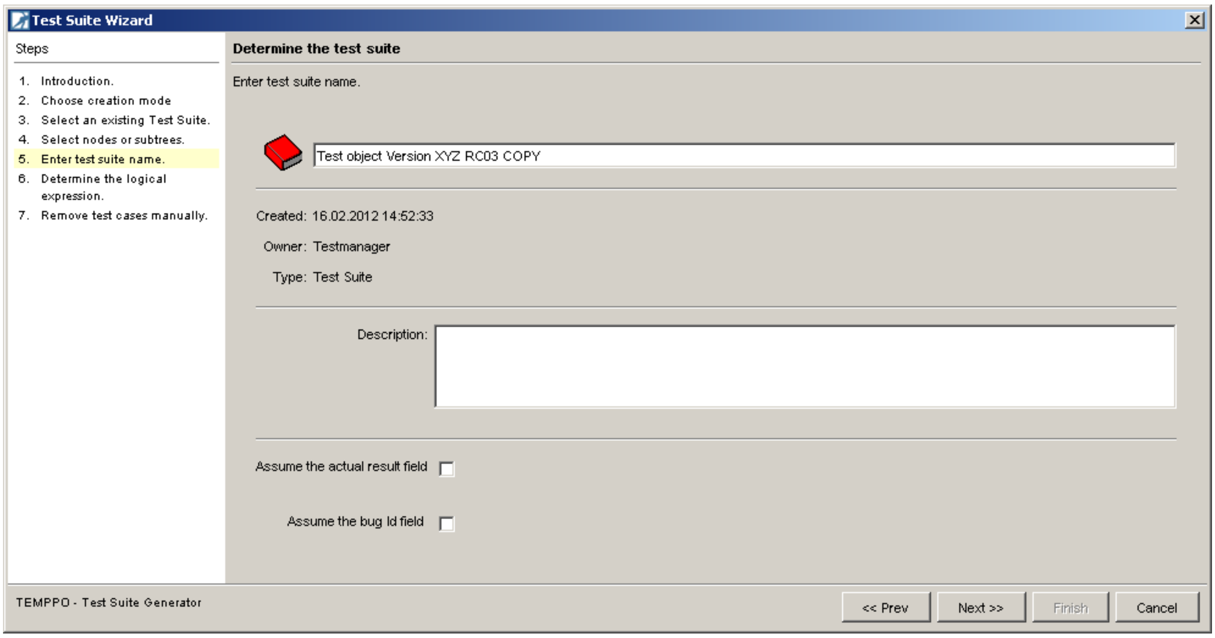
In the first step the test suite tree is shown (Figure 80), where you can select an existing test suite, that will be copied to a new branch.



Test Suite Wizard
Steps
1. Introduction.
2. Choose creation mode
3. Select an existing Test Suite.
Select Test Suite
Select an existing Test Suite.
Notepad - RTF Import with TCs
V1
V1.1 Test Suites for V1.1, 2012-02-16
V1.1.1 editable version
Test object Version XYZ. RC01 [16.02.2012]
Test object Version XYZ RC02 [16.02.2012]
Test object Version XYZ RC03 [16.02.2012]
V2 editable version
TEMPPO - Test Suite Generator
<< Prev
Next >>
Finish
Cancel

Figure 80 - Copy: Select a test suite

In the final step test suite **name** and **comment** are to be specified. Additionally you can configure, if you want to get **Actual Result** and **Bug ID** of source test suite. You can retest bug fix more efficiently.


Test Suite Wizard
Steps
1. Introduction.
2. Choose creation mode
3. Select an existing Test Suite.
4. Select nodes or subtrees.
5. Enter test suite name.
6. Determine the logical expression.
7. Remove test cases manually.
TEMPPO - Test Suite Generator
<< Prev
Next >>
Finish
Cancel

Determine the test suite

Enter test suite name.

Test object Version XYZ RC03 COPY

Created: 16.02.2012 14:52:33

Owner: Testmanager

Type: Test Suite

Description:

Assume the actual result field □

Assume the bug Id field □

Figure 81 - Copy: Enter name and comment

3.1.4.1.4Special: Based on an existing test suite with restrictions to test results

This mode is a special one that solves a problem that arises in many projects. Normally after test case execution you'll have a test suite with passed and failed test cases and therefore bugs in the SUT that have to be fixed in future versions. For testing such a new version, it can be decided to test only the test cases that failed in the previous test execution.

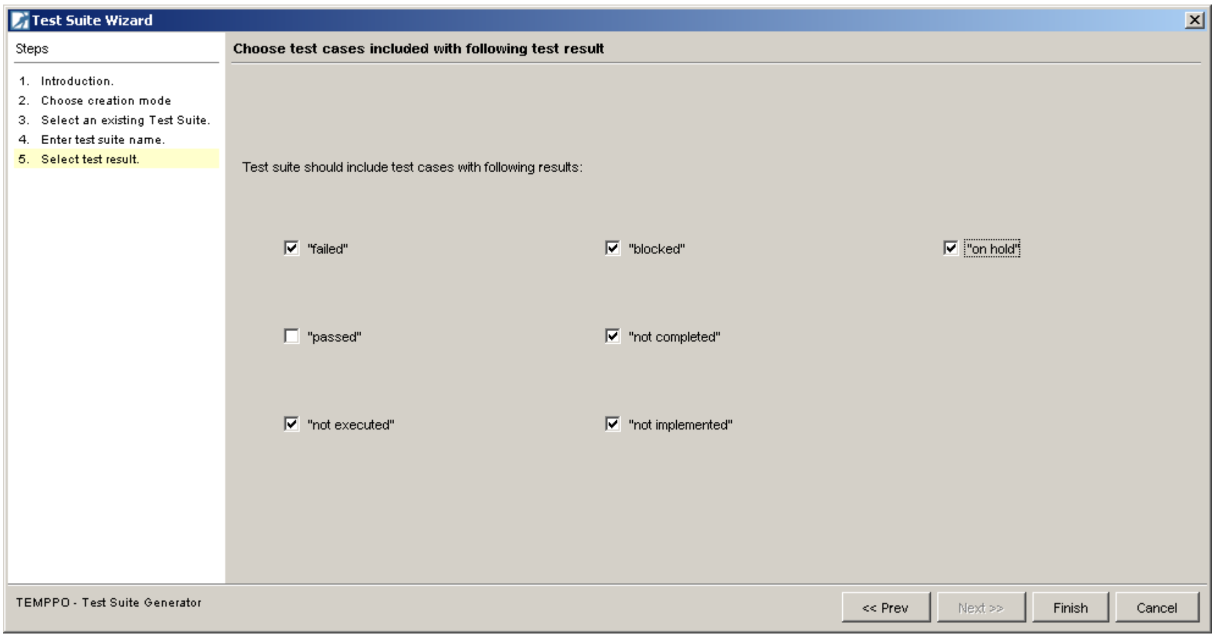
In TEMPPO you are supported by the special mode that provides you to copy a certain test suite with the restriction of test results.

The following steps are to do:

- **Select an existing test suite**
- **Test suite name**
- **Test cases with special results**

For the first 2 steps see 3.1.4.1.3.

In the 3rd step you can select test case results that will filter the test suite. Only the selected ones will be in the new test suite.



Test Suite Wizard

Steps

1. Introduction.
2. Choose creation mode
3. Select an existing Test Suite.
4. Enter test suite name.
5. Select test result.

Choose test cases included with following test result

Test suite should include test cases with following results:

☑ "failed" ☑ "blocked" ☑ "on hold"

☐ "passed" ☑ "not completed"

☑ "not executed" ☑ "not implemented"

TEMPPO - Test Suite Generator

Figure 82 - Special: Restrict test suite

ⓘ **Newly created test cases are added to restricted test suites.**

Consider the following situation: you activated the latest test structure version and created a new manual test case. Now you create a new test suite based on an existing one with the logical expression: type=manual. In step 5 of the wizard choose

Test suite should include all test cases except those which are passed.

Now, the new test suite will not only contain the "failed" test cases of the previous test suite, but also the newly created test case, although its result is "not executed"!

3.1.4.1.5 Manual Test Execution

For every test case, you process its test steps. The sequence of test cases and test steps is arbitrary, you may proceed in any order that suites your needs best. For easier navigation from one test case to another, the buttons on top of the test suite tree can be used (first, previous, next, and last).

A Test Step can be given the following results.

▬ not executed: the test step hasn't been executed yet (initial result)

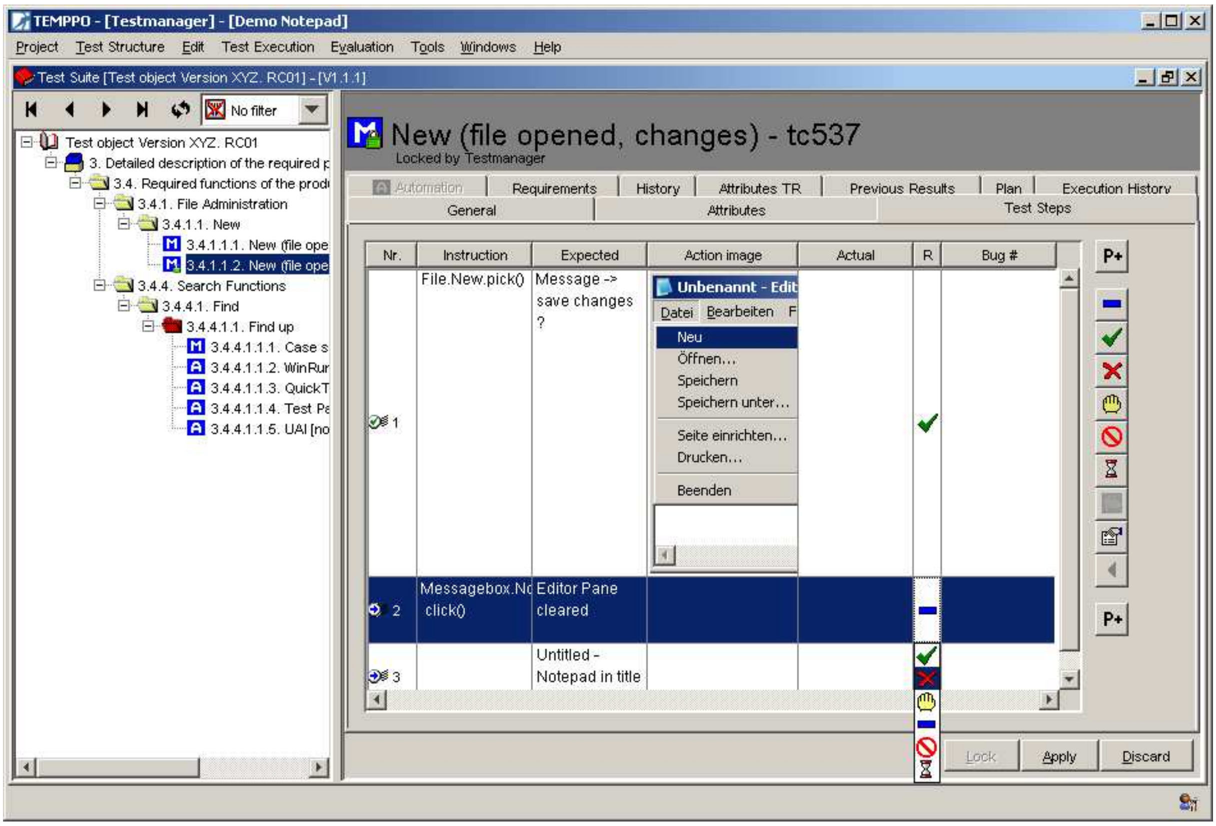
✓ passed: the actual behavior matches the expected one

✗ failed: the actual behavior does not match the expected one

✋ blocked: the test step couldn't be executed, because some pre-condition failed

🚫 not implemented the test step couldn't be executed, because the SW part to be tested is not implemented.

⌛ on hold the test step couldn't be executed, because the SW part to be tested is postponed to another milestone.



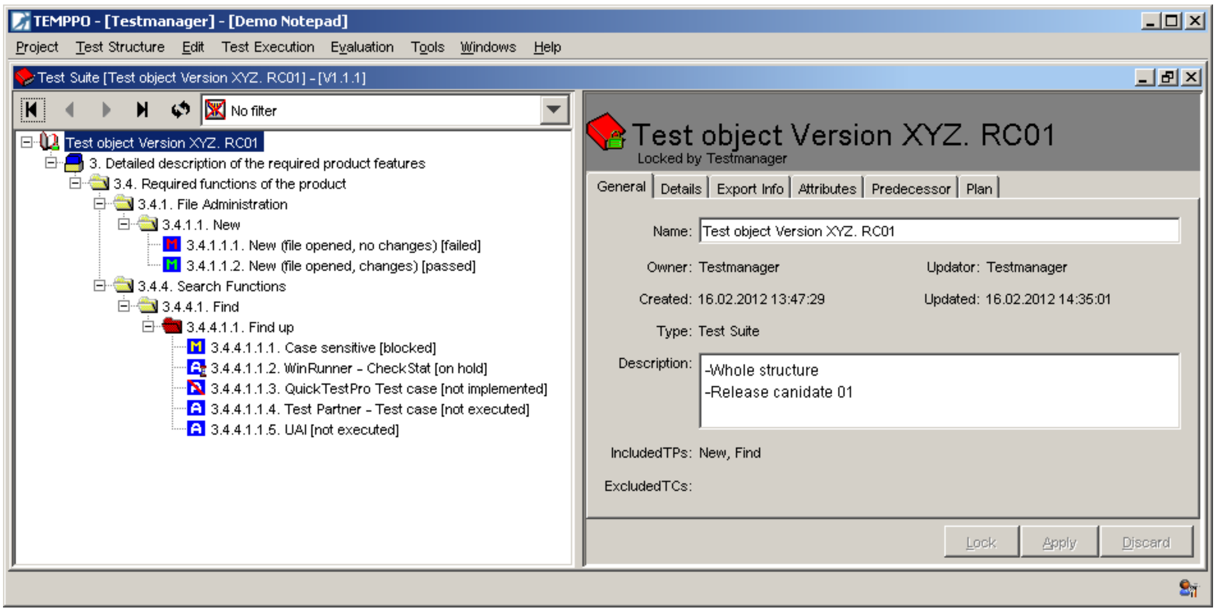
Nr.	Instruction	Expected	Action image	Actual	R	Bug #
1	File.New.pick()	Message -> save changes ?			✓	
2	Messagebox.No click()	Editor Pane cleared				
3		Untitled - Notepad in title				

Figure 83 - Manual Test Execution, Test Step Results

Test case results are calculated by accumulating test step results. So a test case is

- not executed, if all test steps are not executed (initial result).
- passed, if all test steps are passed.
- failed, if at least one test step failed.
- blocked, if no test step failed and at least one is blocked.
- not completed, if no test step failed or is blocked and at least one is passed.
- not implemented, if at least one test step is not implemented (stronger than "on hold").
- on hold, if at least one test step is on hold.

Test case results are displayed within square brackets following the test case name.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
No filter
General Details Export Info Attributes Predecessor Plan
Lock Apply Discard

Test Suite [Test object Version XYZ. RC01] - [V1.1.1]

- Test object Version XYZ. RC01
  - 3. Detailed description of the required product features
    - 3.4. Required functions of the product
      - 3.4.1. File Administration
        - 3.4.1.1. New
          - 3.4.1.1.1. New (file opened, no changes) [failed]
          - 3.4.1.1.2. New (file opened, changes) [passed]
      - 3.4.4. Search Functions
        - 3.4.4.1. Find
          - 3.4.4.1.1. Find up
            - 3.4.4.1.1.1. Case sensitive [blocked]
            - 3.4.4.1.1.2. WinRunner - CheckStat [on hold]
            - 3.4.4.1.1.3. QuickTestPro Test case [not implemented]
            - 3.4.4.1.1.4. Test Partner - Test case [not executed]
            - 3.4.4.1.1.5. UAI [not executed]

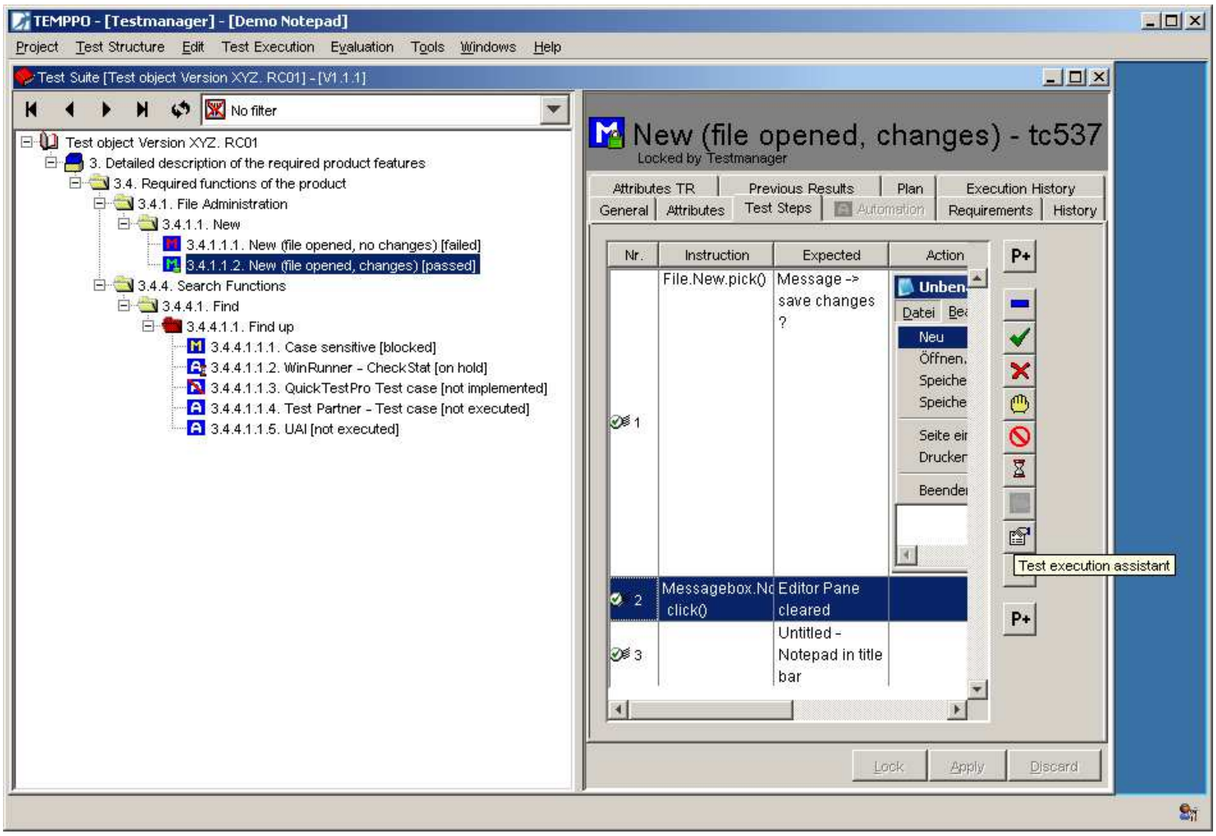
Test object Version XYZ. RC01
Locked by Testmanager

Name: Test object Version XYZ. RC01
Owner: Testmanager Updator: Testmanager
Created: 16.02.2012 13:47:29 Updated: 16.02.2012 14:35:01
Type: Test Suite
Description: -Whole structure
-Release canidate 01
IncludedTPs: New, Find
ExcludedTCs:

Figure 84 - Test Execution, Test Case Results

3.1.4.1.5.1 Test execution assistant

Test execution assistant supports you in executing each step by displaying each step in an own window. It can be called by selecting a step and pressing the button **Test Execution Assistant.**


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
No filter
Attributes TR Previous Results Plan Execution History
General Attributes Test Steps Automation Requirements History
P+
Test execution assistant
Lock Apply Discard
Unben
Datei Bea
Neu
Öffnen.
Speiche
Speiche
Seite eir
Drucker
Beende

Test Suite [Test object Version XYZ. RC01] - [V1.1.1]

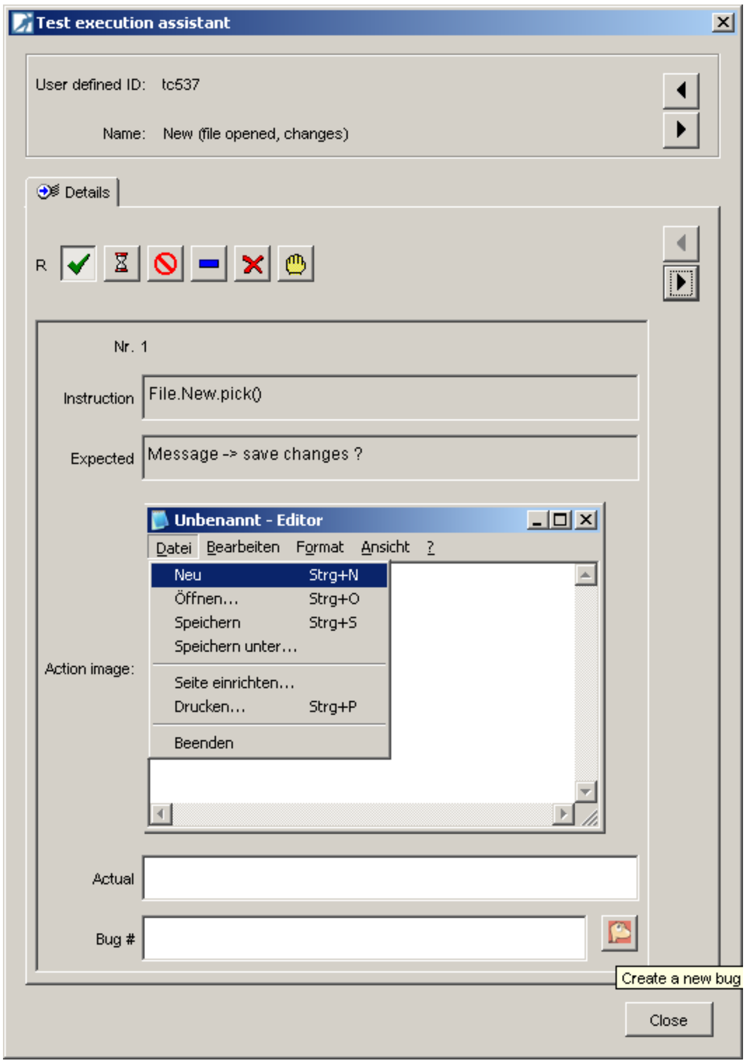
- Test object Version XYZ. RC01
  - 3. Detailed description of the required product features
    - 3.4. Required functions of the product
      - 3.4.1. File Administration
        - 3.4.1.1. New
          - 3.4.1.1.1. New (file opened, no changes) [failed]
          - 3.4.1.1.2. New (file opened, changes) [passed]
      - 3.4.4. Search Functions
        - 3.4.4.1. Find
          - 3.4.4.1.1. Find up
            - 3.4.4.1.1.1. Case sensitive [blocked]
            - 3.4.4.1.1.2. WinRunner - CheckStat [on hold]
            - 3.4.4.1.1.3. QuickTestPro Test case [not implemented]
            - 3.4.4.1.1.4. Test Partner - Test case [not executed]
            - 3.4.4.1.1.5. UAI [not executed]

New (file opened, changes) - tc537
Locked by Testmanager

Nr.	Instruction	Expected	Action
1	File.New.pick()	Message -> save changes ?	
2	Messagebox.No click()	Editor Pane cleared	
3		Untitled - Notepad in title bar	

Figure 85 - Test Execution

Test execution assistant shows all the fields of a test step and allows the user to enter an actual result as well as a bug ID. By pressing one of arrow buttons user can navigate to next or previous test step.



Test execution assistant

User defined ID: tc537

Name: New (file opened, changes)

Nr. 1

Instruction: File.New.pick()

Expected: Message -> save changes ?

Action image:

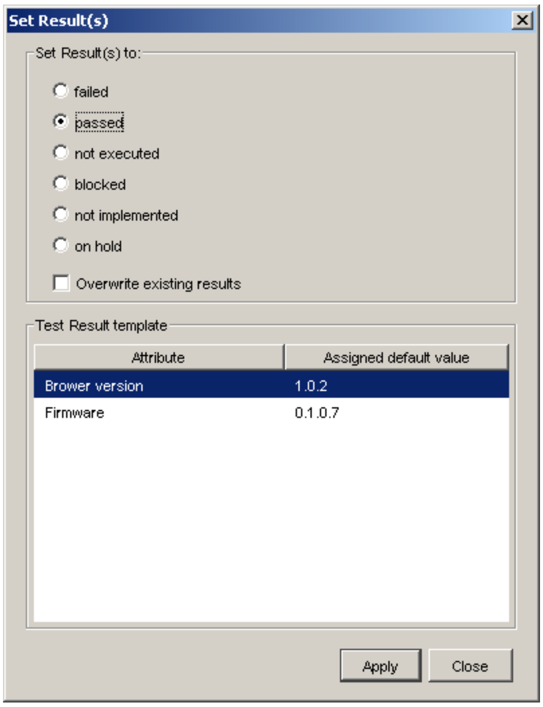
Actual:

Bug #:

Figure 86 - Test Execution, Test Execution Assistant

3.1.4.1.5.2 Set result(s)

With context menu **Set result(s)** it is possible to set test case results for multi selected test cases. Additionally you can set values for test result attributes, if test result template is active.



Set Result(s)

Set Result(s) to:

○ failed
● passed
○ not executed
○ blocked
○ not implemented
○ on hold

□ Overwrite existing results

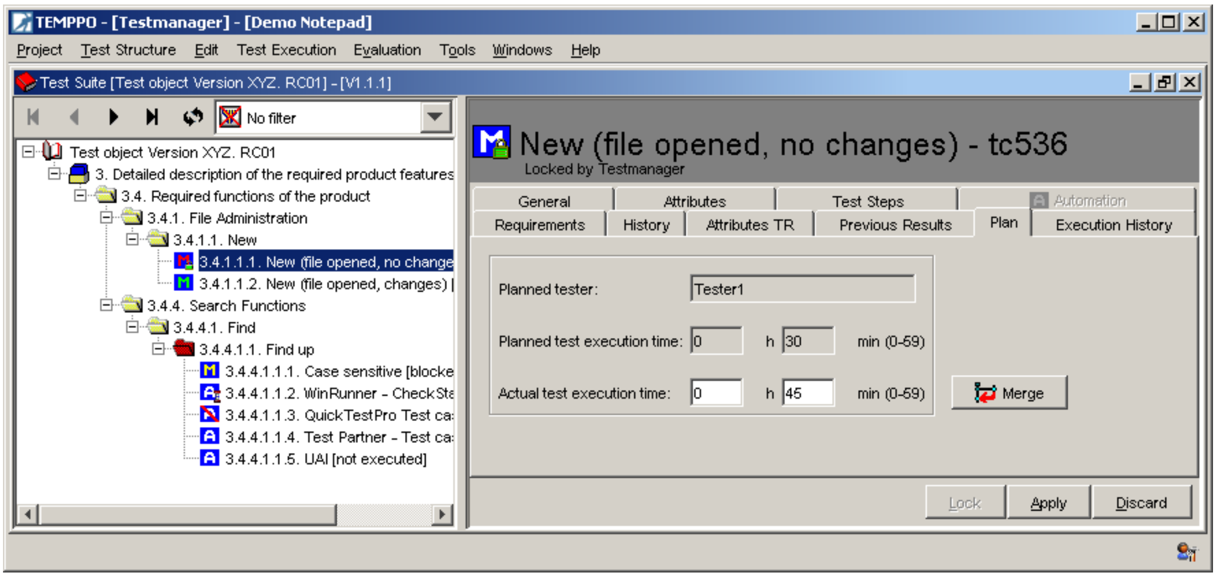
Test Result template

Attribute	Assigned default value
Brower version	1.0.2
Firmware	0.1.0.7

Figure 87 - Set result(s)

3.1.4.1.6 Plan

In test structure test designers estimate the **Planned test execution time**. After test case execution testers enter the **Actual test execution time.** If this time is different to the plan value you can apply that value to the latest version as new **Planned test execution time** by pressing the **Merge** button.



TEMPPO - [Testmanager] - [Demo Notepad]

Test Suite [Test object Version XYZ. RC01] - [V1.1.1]

```
Test object Version XYZ. RC01
  3. Detailed description of the required product features
    3.4. Required functions of the product
      3.4.1. File Administration
        3.4.1.1. New
          3.4.1.1.1. New (file opened, no change
          3.4.1.1.2. New (file opened, changes) [
      3.4.4. Search Functions
        3.4.4.1. Find
          3.4.4.1.1. Find up
            3.4.4.1.1.1. Case sensitive [blocke
            3.4.4.1.1.2. WinRunner - CheckSta
            3.4.4.1.1.3. QuickTestPro Test ca
            3.4.4.1.1.4. Test Partner - Test ca
            3.4.4.1.1.5. UAI [not executed]
```

New (file opened, no changes) - tc536
Locked by Testmanager

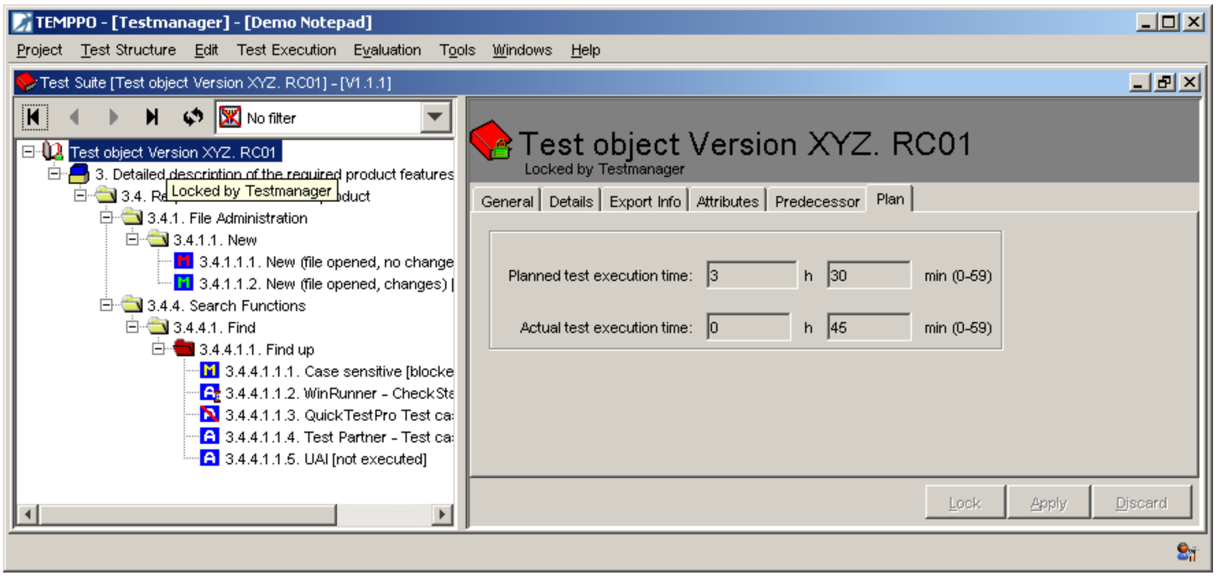
Planned tester: Tester1

Planned test execution time: 0 h 30 min (0-59)

Actual test execution time: 0 h 45 min (0-59)

Figure 88 - Test suite – Test Case - Plan

That information is not only available for test cases but is also viewed for test packages and the test suite itself. When clicking on root node test managers can view the accumulated values for **Planned test execution time** and **Actual test execution time** for whole test suite.


TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Suite [Test object Version XYZ. RC01] - [V1.1.1]
No filter
Test object Version XYZ. RC01
3. Detailed description of the required product features
3.4. Re... Locked by Testmanager ...duct
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no change
3.4.1.1.2. New (file opened, changes) |
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case sensitive [blocke
3.4.4.1.1.2. WinRunner - CheckSta
3.4.4.1.1.3. QuickTestPro Test ca
3.4.4.1.1.4. Test Partner - Test ca
3.4.4.1.1.5. UAI [not executed]
Test object Version XYZ. RC01
Locked by Testmanager
General
Details
Export Info
Attributes
Predecessor
Plan
Lock
Apply
Discard

Planned test execution time: 3 h 30 min (0-59)

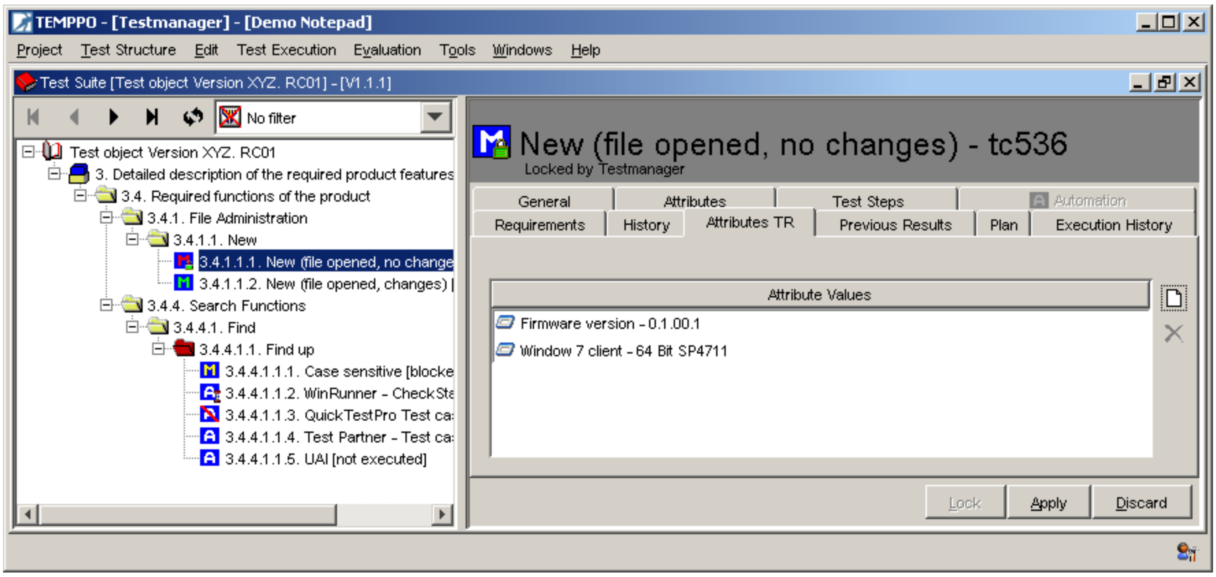
Actual test execution time: 0 h 45 min (0-59)

Figure 89 - Test suite - Plan

3.1.4.1.7Test Result Attributes

There is the possibility to assign attributes to test cases for execution in tab **Attributes TR** of a test case result.

Only attributes can be chosen that are indicated as test result attributes in the Meta Data Editor (see **5.3**).

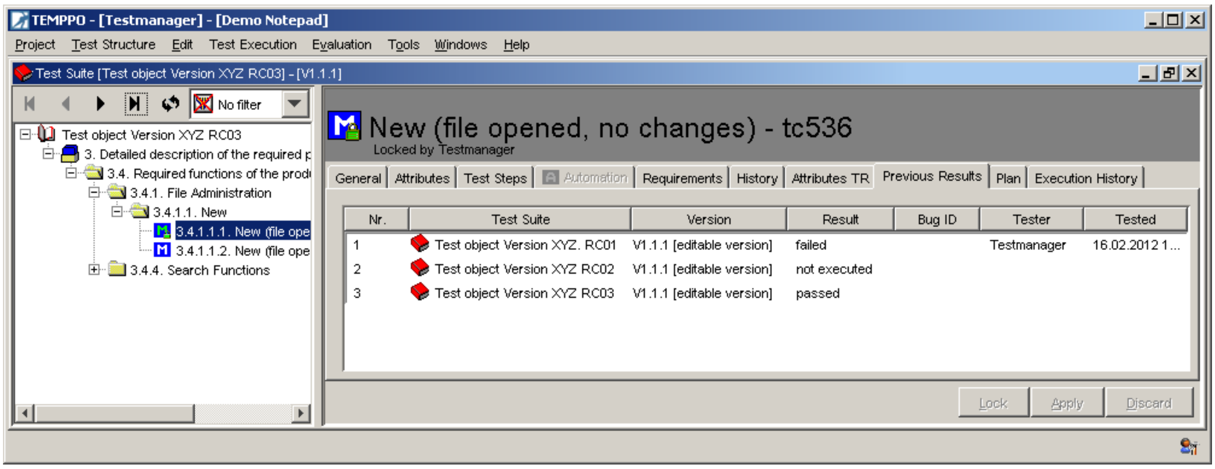

TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Suite [Test object Version XYZ. RC01] - [V1.1.1]
No filter
Test object Version XYZ. RC01
3. Detailed description of the required product features
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no change
3.4.1.1.2. New (file opened, changes) |
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case sensitive [blocke
3.4.4.1.1.2. WinRunner - CheckSta
3.4.4.1.1.3. QuickTestPro Test ca
3.4.4.1.1.4. Test Partner - Test ca
3.4.4.1.1.5. UAI [not executed]
New (file opened, no changes) - tc536
Locked by Testmanager
General
Attributes
Test Steps
Automation
Requirements
History
Attributes TR
Previous Results
Plan
Execution History
Lock
Apply
Discard

Attribute Values
Firmware version - 0.1.00.1
Window 7 client - 64 Bit SP4711

Figure 90 – Test Case Result Attributes

3.1.4.1.8Previous Results

With predecessors, there is also the possibility to get known about the results in the predecessor test suites. The table isn't changeable.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Suite [Test object Version XYZ RC03] - [V1.1.1]
No filter
Test object Version XYZ RC03
3. Detailed description of the required p
3.4. Required functions of the prodi
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file ope
3.4.1.1.2. New (file ope
3.4.4. Search Functions
General
Attributes
Test Steps
Automation
Requirements
History
Attributes TR
Previous Results
Plan
Execution History
Lock
Apply
Discard

New (file opened, no changes) - tc536
Locked by Testmanager

Nr.	Test Suite	Version	Result	Bug ID	Tester	Tested
1	Test object Version XYZ. RC01	V1.1.1 [editable version]	failed		Testmanager	16.02.2012 1...
2	Test object Version XYZ RC02	V1.1.1 [editable version]	not executed			
3	Test object Version XYZ RC03	V1.1.1 [editable version]	passed			

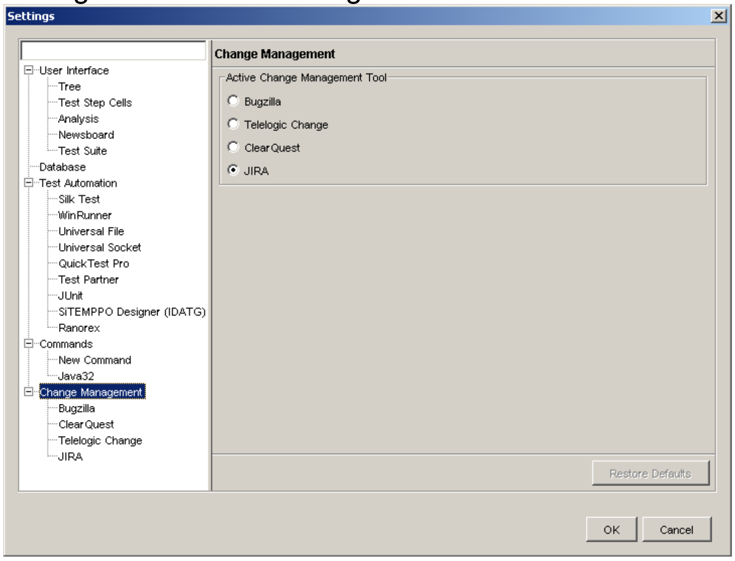
Figure 91 – Previous Results

3.1.4.1.9 Enter new bug entry

An actual behavior, and in case of errors a bug ID can be recorded for each test step and for each automated test case, too.

The bug ID would point to a problem report in a bug tracking tool like **Bugzilla**, **IBM Telelogic Change, IBM Rational Clearquest** or **JIRA**.

Before you can generate a bug automatically, you have to configure the corresponding settings in TEMPPO settings.



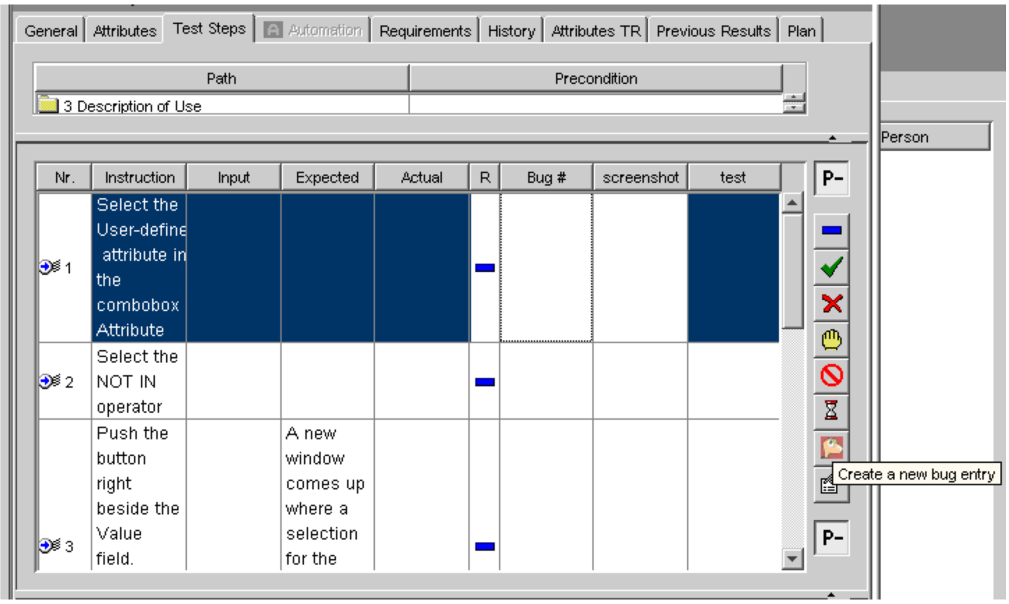
Change Management

Active Change Management Tool

○ Bugzilla
○ Telelogic Change
○ ClearQuest
● JIRA

Figure 92 - TEMPPO settings

For creating a new bug select the corresponding cell (bug ID, step#) and press the button **Create a new bug entry** (see Figure 93).



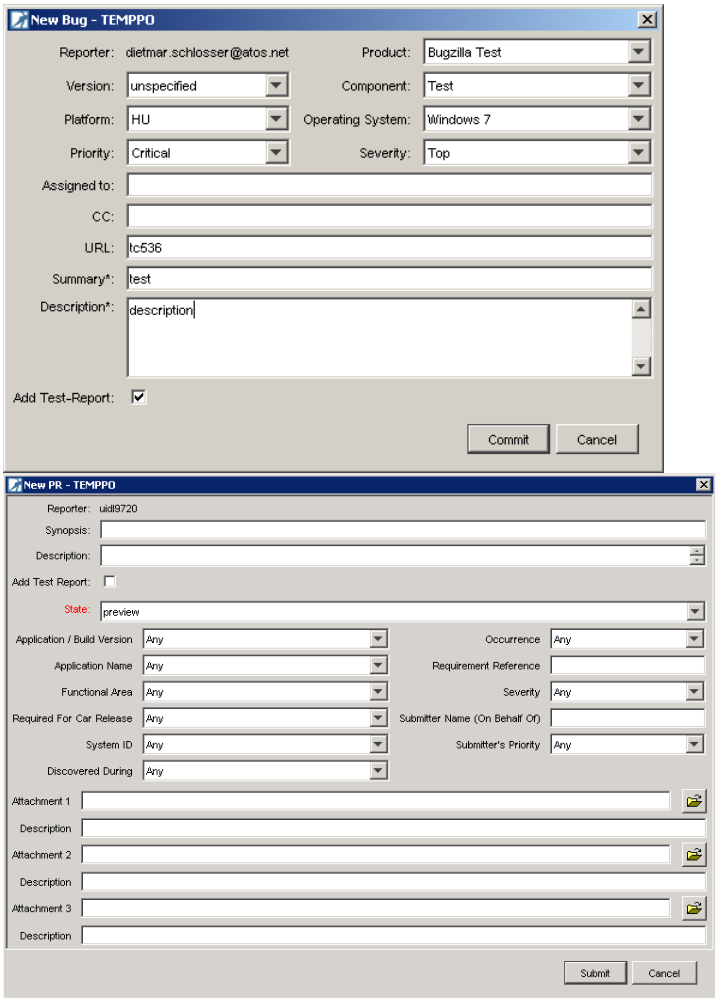
Path	Precondition
3 Description of Use	

Nr.	Instruction	Input	Expected	Actual	R	Bug #	screenshot	test
1	Select the User-define attribute in the combobox Attribute				▬			
2	Select the NOT IN operator				▬			
3	Push the button right beside the Value field.		A new window comes up where a selection for the		▬			

Figure 93 – Create a new bug entry

Now a dialog for creating a bug or change request opens (see Figure 94). Fill in the data of the bug and may activate the combo box **Add test report**. In the end you press **Submit**.

The panel for Telelogic Change is displayed dynamically. So the list boxes beneath the box **State** are dependent on the selected **State**. All fields at the Telelogic Change window are mandatory for the change request.



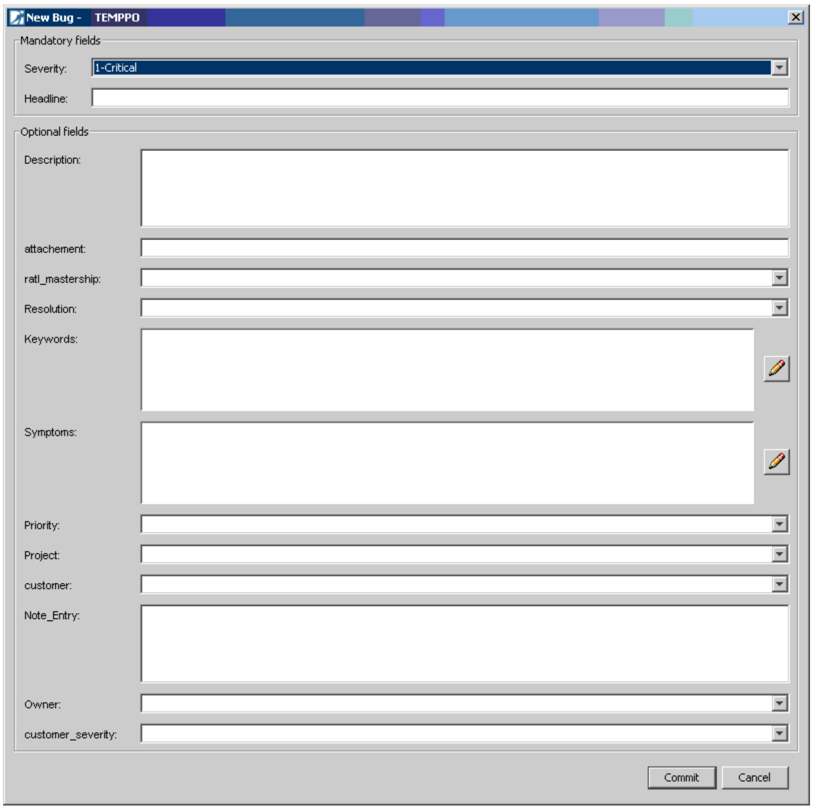
New Bug - TEMPPO

Reporter: dietmar.schlosser@atos.net
Product: Bugzilla Test
Version: unspecified
Component: Test
Platform: HU
Operating System: Windows 7
Priority: Critical
Severity: Top
Assigned to:
CC:
URL: tc536
Summary*: test
Description*: description
Add Test-Report: ☑

New PR - TEMPPO

Reporter: uidl9720
Synopsis:
Description:
Add Test Report: □
State: preview
Application / Build Version: Any
Occurrence: Any
Application Name: Any
Requirement Reference:
Functional Area: Any
Severity: Any
Required For Car Release: Any
Submitter Name (On Behalf Of):
System ID: Any
Submitter's Priority: Any
Discovered During: Any
Attachment 1
Description
Attachment 2
Description
Attachment 3
Description

Figure 94 – Bugzilla / Telelogic Change



New Bug - TEMPPO

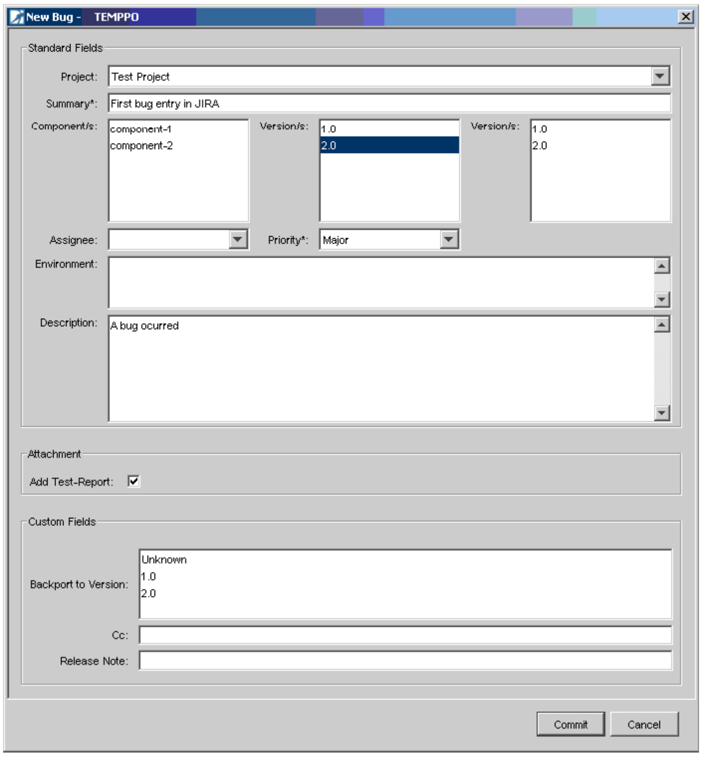
Mandatory fields
Severity: 1-Critical
Headline:

Optional fields
Description:
attachement:
ratl_mastership:
Resolution:
Keywords:
Symptoms:
Priority:
Project:
customer:
Note_Entry:
Owner:
customer_severity:

Figure 95 – Clearquest

Now the report is added to the tracking tool and in the column `Bug ID` you find the hyperlink to your report (see Figure 97).

ⓘ **Limitation: With IBM Rational Clearquest's interface it is not possible to attach directly the test report**



New Bug - TEMPPO

Standard Fields

Project: Test Project

Summary*: First bug entry in JIRA

Component/s: component-1, component-2

Version/s: 1.0, 2.0 (2.0 selected)

Version/s: 1.0, 2.0

Assignee:

Priority*: Major

Environment:

Description: A bug ocurred

Attachment

Add Test-Report: ☑

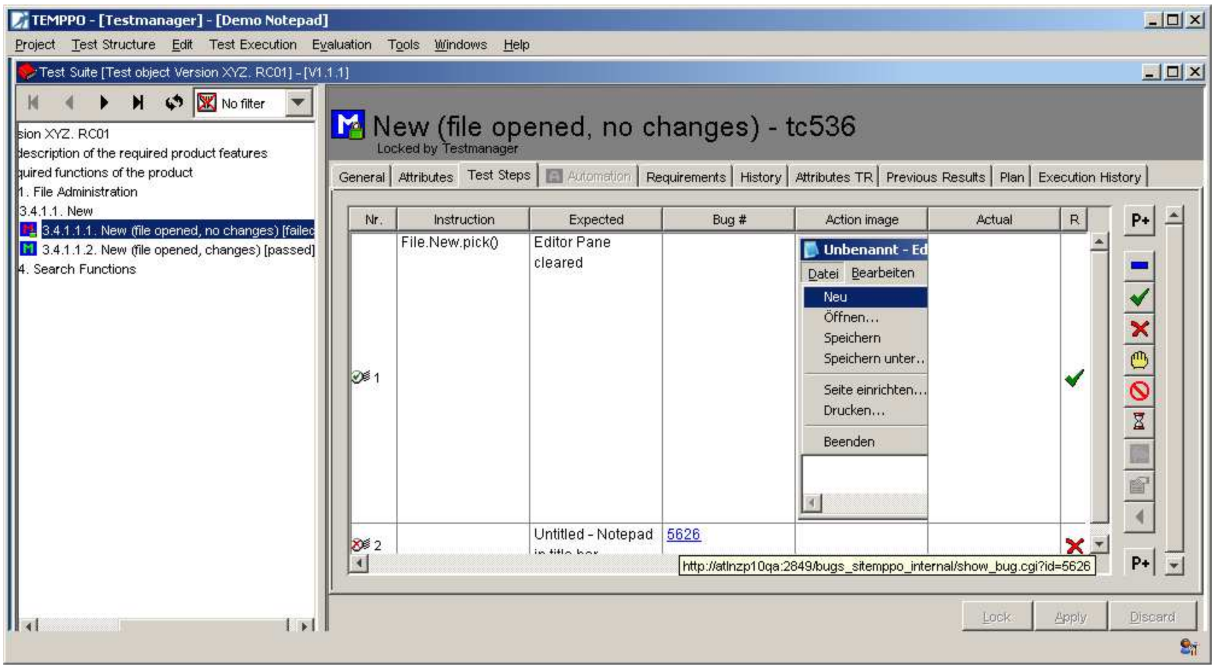
Custom Fields

Backport to Version: Unknown, 1.0, 2.0

Cc:

Release Note:

Figure 96 - JIRA



TEMPPO - [Testmanager] - [Demo Notepad]

Test Suite [Test object Version XYZ. RC01] - [V1.1.1]

sion XYZ. RC01
description of the required product features
quired functions of the product
1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no changes) [failed
3.4.1.1.2. New (file opened, changes) [passed]
4. Search Functions

New (file opened, no changes) - tc536
Locked by Testmanager

Nr.	Instruction	Expected	Bug #	Action image	Actual	R
1	File.New.pick()	Editor Pane cleared				✓
2		Untitled - Notepad in title bar	5626			✗

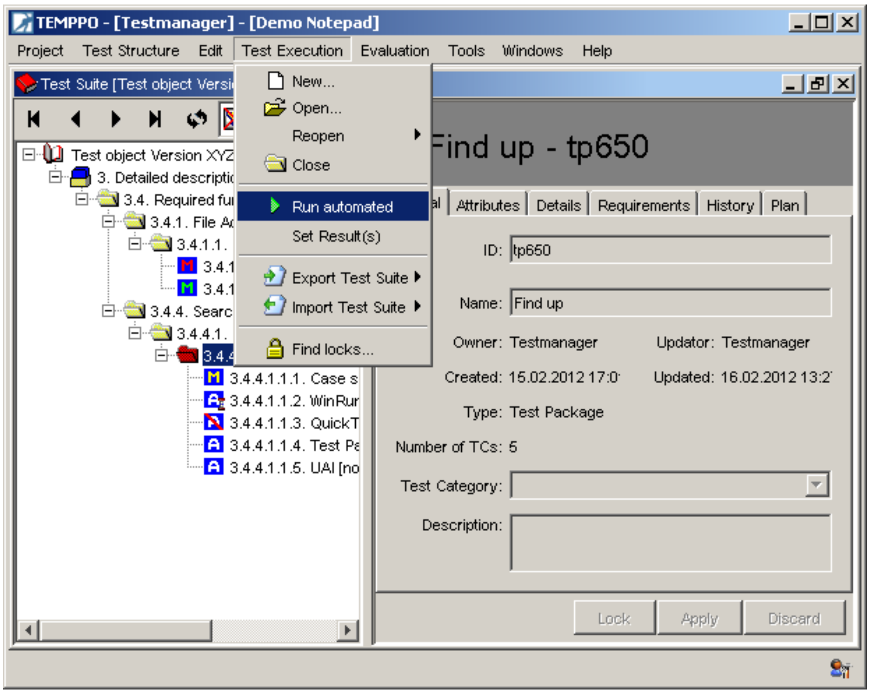
http://atlnzp10qa:2849/bugs_sitemppo_internal/show_bug.cgi?id=5626

Figure 97 – Hyperlink to bug

3.1.4.1.10 Automated Test Execution

For starting an automated test execution, you have to select the test package, which contains the test cases to run, and activate **Test Execution – Run automated** (or the corresponding item in the context menu – see Figure 98). TEMPPO Test Manager will then sequentially invoke the proper automation tool for each test case and fetch the result obtained by the tool after the automated run.

ⓘ **Before running automated tests, the automation settings in TEMPPO's administration tool (see 4) must be configured.**



ID: tp650

Name: Find up

Owner: Testmanager Updator: Testmanager

Created: 15.02.2012 17:0 Updated: 16.02.2012 13:2

Type: Test Package

Number of TCs: 5

Test Category:

Description:

Figure 98 - Automated Test Execution

3.1.4.1.11 Mixed Test Execution

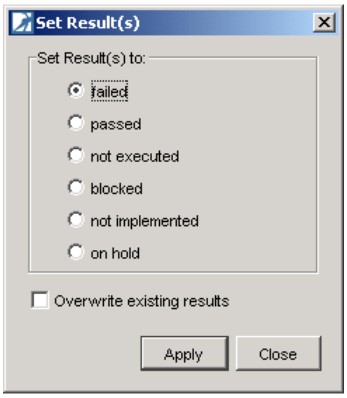
A mixed test suite contains both manual and automated test cases. The test execution doesn't differ from the one described above, but is just a mixture of them.

If a test package contains both manual and automated test cases and **Run automated** is invoked on it, only the automated test cases will be executed. All manual test cases in this test package must be carried out afterwards.

3.1.4.1.12 Setting several results

For regression testing it is often convenient to set results for several test cases at once; e.g. if you already know the results of the test cases. First you have to select a (sub) tree in the test suite and then activate the (context) menu **Set result(s)**, and afterwards Figure 99 is shown. You can choose one result and optionally set the checkbox **Overwrite existing results** and after clicking Apply all test results are set.

ⓘ **If other users have locked test cases, it is not possible to set the result, of course. You will be informed if results couldn't be set.**



Set Result(s)

Set Result(s) to:

● failed
○ passed
○ not executed
○ blocked
○ not implemented
○ on hold

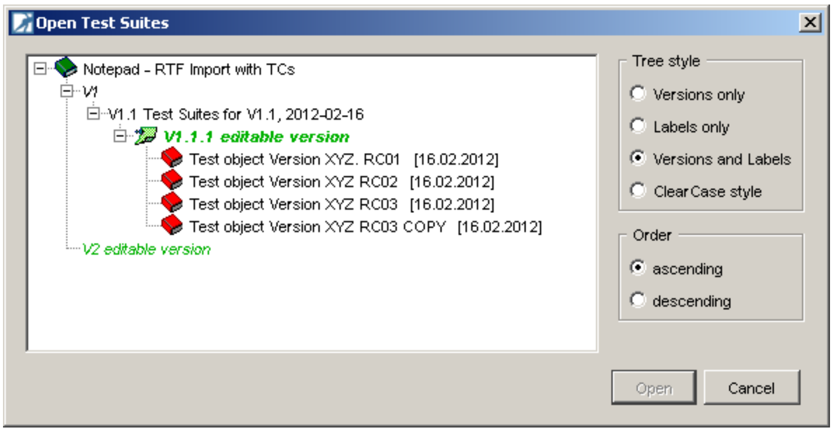
☐ Overwrite existing results

Figure 99 - Set Result(s)

ⓘ **If a test case contains no test steps, it remains "not executed".**

3.1.4.2 Pausing and Continuing Test Execution

If there are a lot of test cases, you will want to pause test execution and continue at a later time. TEMPPO Test Manager allows you to close your test suite any time you like. Later on, you can reopen it by activating **Test Execution > Open**. The dialog shown in Figure 100 will be displayed. This dialog also shows the version-tree extended by the test suite (📕) beneath the version where it was created. During the opening process, the adequate version of the test structure is loaded automatically.



Open Test Suites

```
⊟ Notepad - RTF Import with TCs
   ⊟ V1
      ⊟ V1.1 Test Suites for V1.1, 2012-02-16
         ⊟ V1.1.1 editable version
               Test object Version XYZ. RC01   [16.02.2012]
               Test object Version XYZ RC02   [16.02.2012]
               Test object Version XYZ RC03   [16.02.2012]
               Test object Version XYZ RC03 COPY   [16.02.2012]
   V2 editable version
```

Tree style
○ Versions only
○ Labels only
● Versions and Labels
○ ClearCase style

Order
● ascending
○ descending

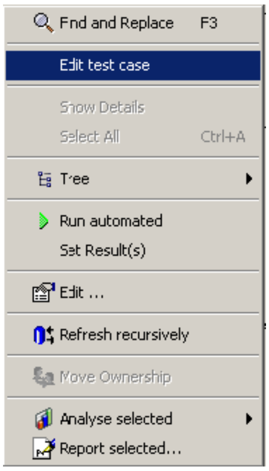
Figure 100 - Open an existing test suite

After the test suite is opened, you can proceed with your test execution as described in the previous chapter. You can repeat this cycle as many times as you like.

3.1.4.3 Editing Test Cases during Test Execution

As stated in the introductory paragraphs of the test execution chapter, it might well be that you detect faults in your test cases just before you want to execute them.

If you want to edit (also cut, copy, paste) test steps before or during test case execution, it is the best way to select the test case and call the menu item **Edit test case**. The test structure is highlighted and the test case is automatically selected.



Then add e.g. a test step (see new test step 4 in Figure 101). The **Apply** and **Discard** buttons become enabled. When pressing the **Apply** button, the changes done in the test structure will be applied to the test suite and you can execute this new or edited test step.

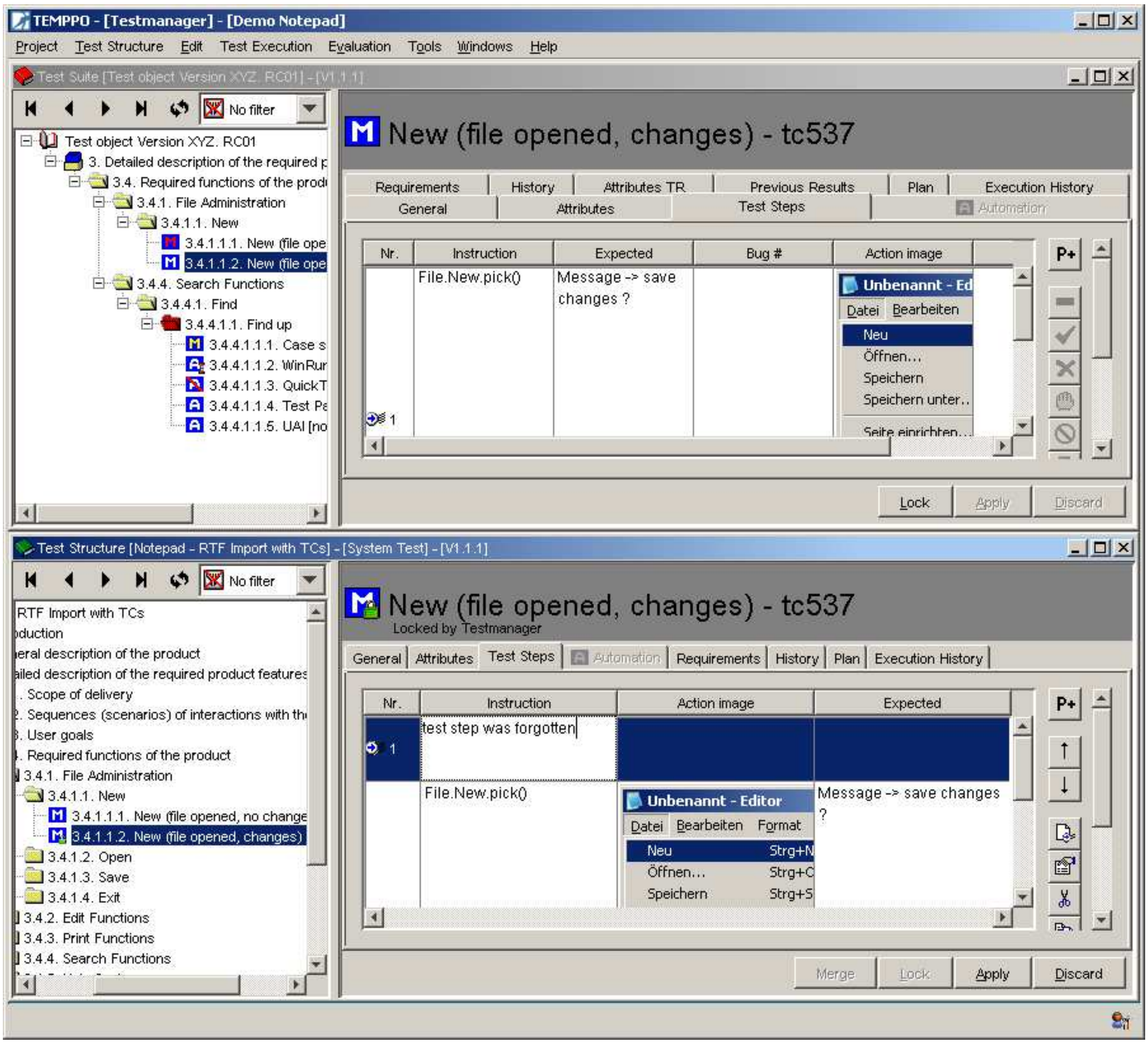
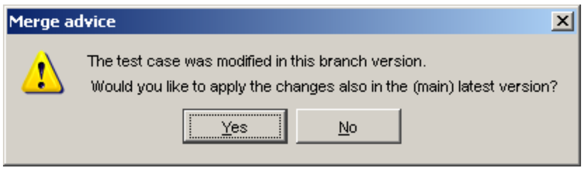

TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Suite [Test object Version XYZ. RC01] - [V1.1.1]
No filter
Test object Version XYZ. RC01
3. Detailed description of the required p
3.4. Required functions of the prod
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file ope
3.4.1.1.2. New (file ope
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case s
3.4.4.1.1.2. WinRur
3.4.4.1.1.3. QuickT
3.4.4.1.1.4. Test Pa
3.4.4.1.1.5. UAI [no
New (file opened, changes) - tc537
Requirements History Attributes TR Previous Results Plan Execution History
General Attributes Test Steps Automation
Nr. Instruction Expected Bug # Action image
1 File.New.pick() Message -> save changes ?
Unbenannt - Ed
Datei Bearbeiten
Neu
Öffnen...
Speichern
Speichern unter..
Seite einrichten...
P+
Lock Apply Discard
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1.1.1]
No filter
RTF Import with TCs
oduction
eral description of the product
ailed description of the required product features
. Scope of delivery
2. Sequences (scenarios) of interactions with th
3. User goals
. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no change
3.4.1.1.2. New (file opened, changes)
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
New (file opened, changes) - tc537
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
Nr. Instruction Action image Expected
1 test step was forgotten
File.New.pick() Message -> save changes ?
Unbenannt - Editor
Datei Bearbeiten Format
Neu Strg+N
Öffnen... Strg+O
Speichern Strg+S
P+
Merge Lock Apply Discard

Figure 101 Editing test cases during test case execution

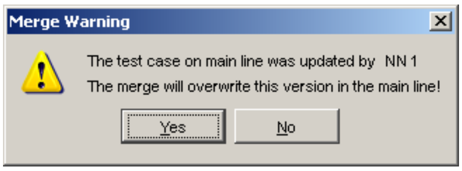
But remember, there still remains the problem that you have edited a test case on the branch. Very likely you will need these changes for all further test executions that will be branched from the (latest) version of the main line. What can you do to solve that problem?

There are two possibilities: On one hand you can merge each single (edited) test case. After pressing the Apply button the user is automatically asked to merge the changes. Remark: The Merge button can always be pressed if enabled.



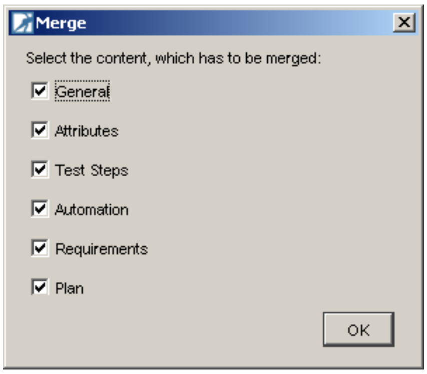
The test case was modified in this branch version.
Would you like to apply the changes also in the (main) latest version?

Additionally TEMPPO Test Manager checks, if the test case to be merged is already changed in the mean time and informs the user:



The test case on main line was updated by NN 1
The merge will overwrite this version in the main line!

If pressing **Yes**, you may select which content should be merged into the latest version, see Figure 102.



Select the content, which has to be merged:

☑ General
☑ Attributes
☑ Test Steps
☑ Automation
☑ Requirements
☑ Plan

Figure 102 - Merge content

The corresponding content is “copied” from the selected branch to the latest version of the main line, which means that the current version of this test case parts (**general**, **attributes**, **test steps**, **automation requirements**, **plan**) in main latest is overwritten.

For advanced TEMPPO users there is a possibility to merge ALL test cases and test packages from a branch or main version to the latest version of the main line (e.g. after finishing test execution and test case updating). You can open the **version** window and press the **Merge all test cases into latest version of main line** button.

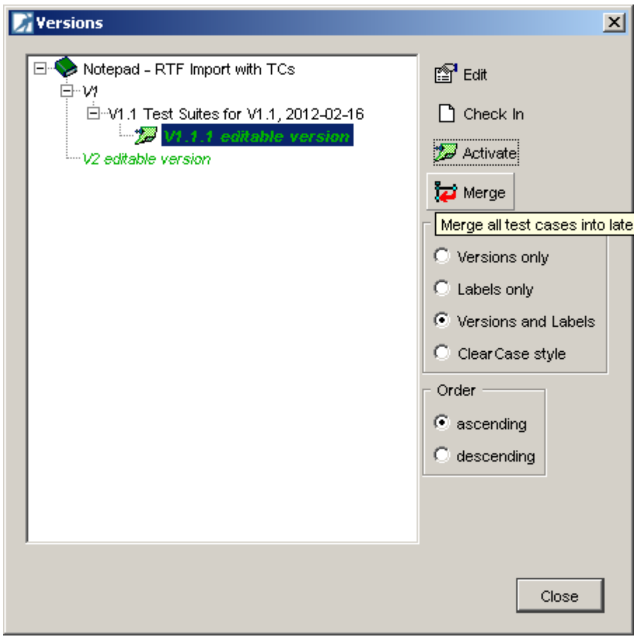


Figure 103 – Versions

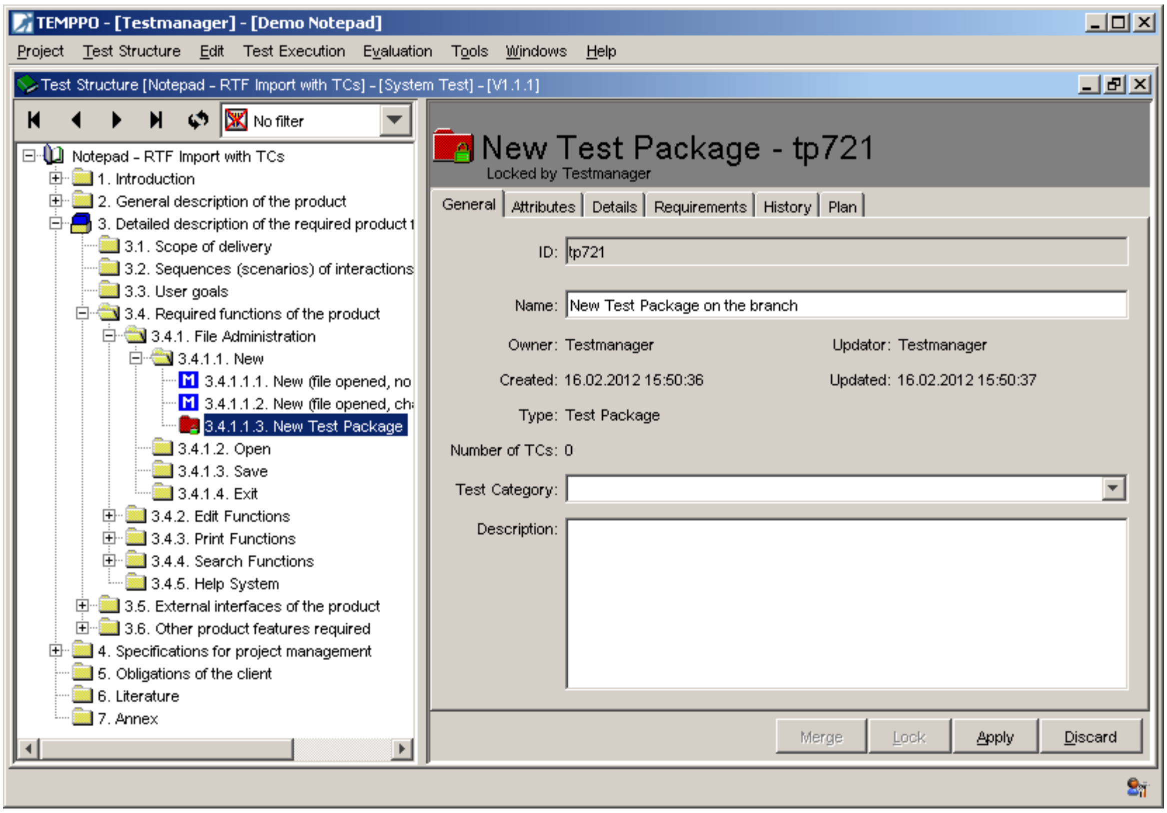
After pressing the merge button, a warning window comes up, that asks you, if you are sure to go on. If you commit this window, the latest version will be checked in and checked out again. Then all test cases of the branch version are copied to this checked out version.

3.1.4.3.1 Adding new test cases during test case execution

As a matter of fact new test cases have to be added to test case execution, although that test phase already started. Therefore such test cases have to be created in the corresponding test structure (branch) versions. If the pre-selection of test packages (sub-trees) and the logical expression when creating the test suite match, the test cases appear in the test suite (after reloading the whole tree).

3.1.4.3.2 Adding and changing test packages during test case execution

During test case execution it might be necessary to create new test packages on a branch (see Figure 104) and use them also in the latest version later on.



TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1.1.1]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required product
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no
3.4.1.1.2. New (file opened, ch
3.4.1.1.3. New Test Package
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.5. Help System
3.5. External interfaces of the product
3.6. Other product features required
4. Specifications for project management
5. Obligations of the client
6. Literature
7. Annex
New Test Package - tp721
Locked by Testmanager
General Attributes Details Requirements History Plan
Merge Lock Apply Discard

ID: tp721

Name: New Test Package on the branch

Owner: Testmanager
Updator: Testmanager

Created: 16.02.2012 15:50:36
Updated: 16.02.2012 15:50:37

Type: Test Package

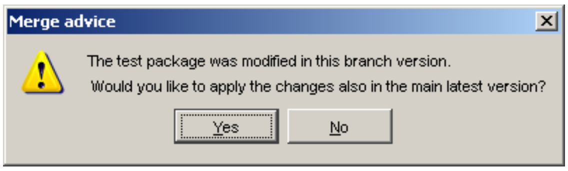
Number of TCs: 0

Test Category:

Description:

Figure 104 - Creating new test package on branch

After pressing the Apply button the user automatically asked if he wants to apply the new test package or changes on it to the latest version.



Merge advice

The test package was modified in this branch version.
Would you like to apply the changes also in the main latest version?

3.1.4.4 Finding Bug IDs

If you have the problem of finding bug IDs and do not remember the test cases, you activate the menu item **Find and Replace**. For more details see 3.3.8.

3.1.4.5 Finishing Test Execution

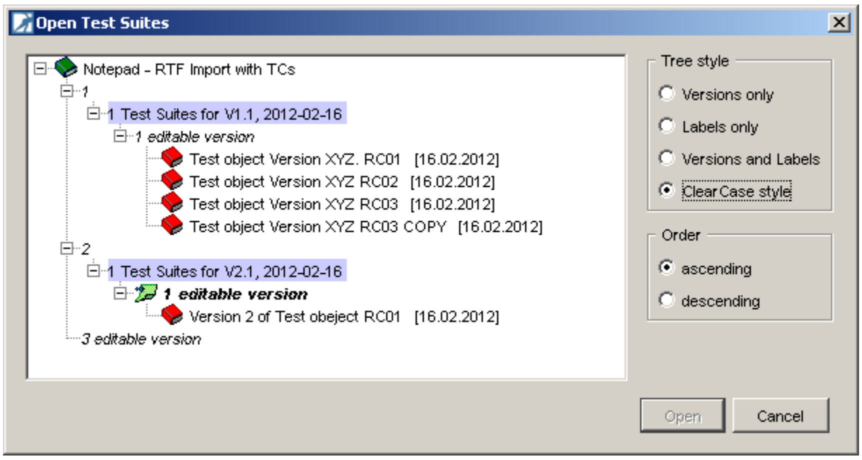
After all of the test cases in your test suite have their results recorded, you have two choices:

check in the current version

leave it checked out
(in this case, TEMPPO Test Manager will prohibit editing test cases, because they have results)

3.1.4.6 Creating More Test Suites

Here is an example with more test suites for more than one application release.



Open Test Suites

```
Notepad - RTF Import with TCs
  1
    1 Test Suites for V1.1, 2012-02-16
      1 editable version
        Test object Version XYZ. RC01   [16.02.2012]
        Test object Version XYZ RC02   [16.02.2012]
        Test object Version XYZ RC03   [16.02.2012]
        Test object Version XYZ RC03 COPY   [16.02.2012]
  2
    1 Test Suites for V2.1, 2012-02-16
      1 editable version
        Version 2 of Test obeject RC01   [16.02.2012]
  3 editable version
```

Tree style
○ Versions only
○ Labels only
○ Versions and Labels
● ClearCase style

Order
● ascending
○ descending

Figure 105 - An extra branch for each test execution

Typically, each test execution has its own branch which is directly beneath the main line and contains just one test suite. This is because faults in test cases are usually corrected in the main line of the version tree (versions *1*, *2*, *3*, *4* in the example). If every new test execution branches directly off the main line, it will always have the most up-to-date definitions of test cases.

ⓘ **Grouping test executions within a common branch is possible, yet not recommended due to the suggested procedure described above. Only highly experienced users should consider departing from this approach.**

3.1.4.7 Test case execution with an exported test suite

In projects for certain domains tests can't be executed in the office on the desktop PC, but must be executed offline (e.g. in a test-car for the automotive domain). For this purpose TEMPPO provides features, which enable:

- to export a test suite into an export database
- execute test cases and store the test results in this export database
- afterwards import the test results into the origin database
- remove a test suite from such an export database

Consider the following situation: You have a test team of 2 or more testers and have a central database. For test case execution it is necessary to unplug the notebook or workstation from the database (LAN) and execute the test cases somewhere else (e.g. in a car).

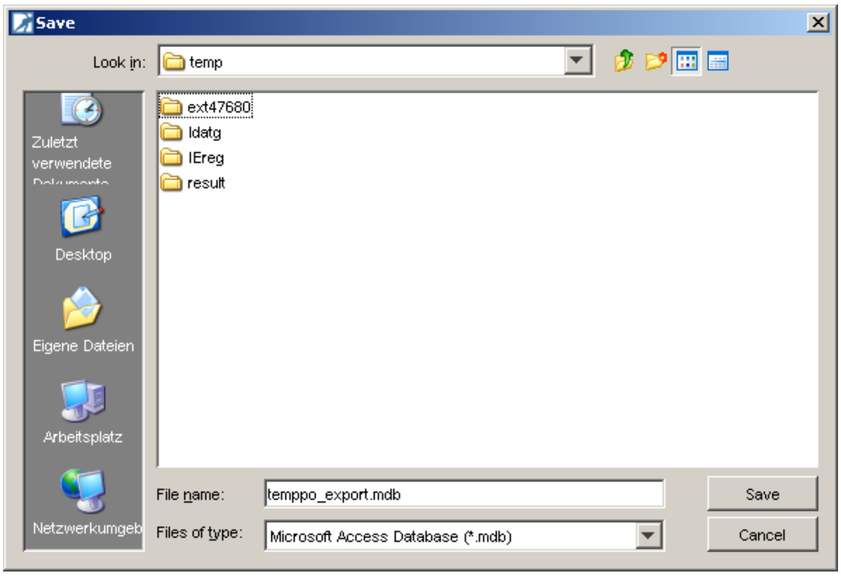
3.1.4.7.1 Export a test suite

First of all you have to be connected to the central database with the TEMPPO Administrator. After starting TEMPPO Test Manager you select project, test structure and open the test suite containing the test cases to be executed offline.

Note that a set filter is concerned for export, which means that only the visible test suite is exported!

When activating the menu item **Export Test Suite** the exporting process to a local MS Access database is started.

At first you select the directory for the export database (see Figure 106). The default database name is **TEMPPO_export.mdb**.



Look in: temp

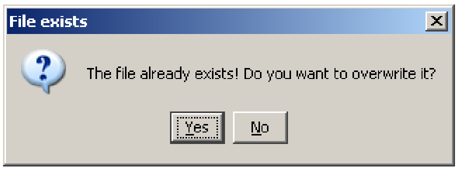
ext47680
ldatg
IEreg
result

File name: temppo_export.mdb

Files of type: Microsoft Access Database (*.mdb)

Figure 106 – Save dialog

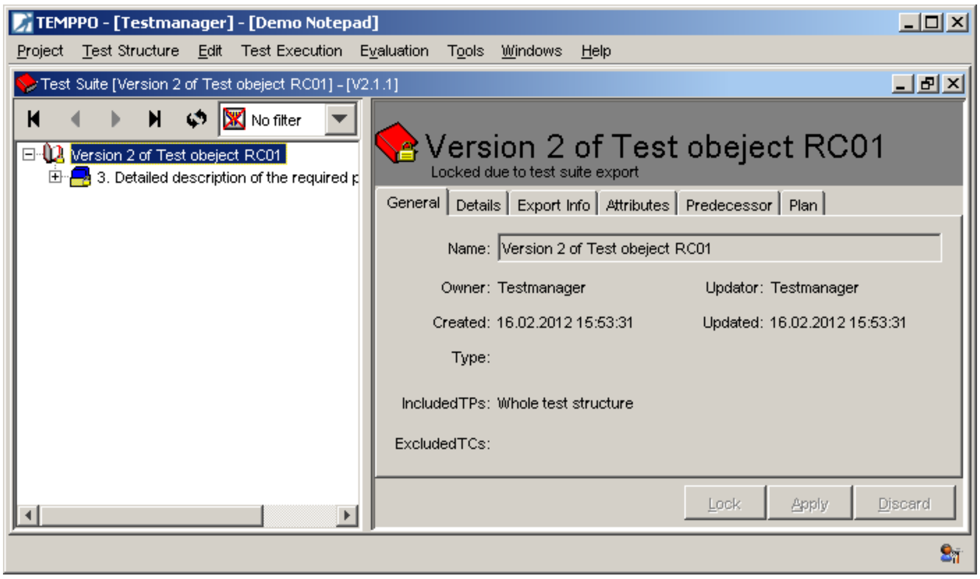
If the export database already exists, you get the following message (Figure 107):



The file already exists! Do you want to overwrite it?

Figure 107 – Overwrite file

After successful export the test suite with activated root node looks as follows:



Test Suite [Version 2 of Test obeject RC01] - [V2.1.1]

Version 2 of Test obeject RC01
3. Detailed description of the required p

Version 2 of Test obeject RC01
Locked due to test suite export

Name: Version 2 of Test obeject RC01

Owner: Testmanager
Updator: Testmanager

Created: 16.02.2012 15:53:31
Updated: 16.02.2012 15:53:31

Type:

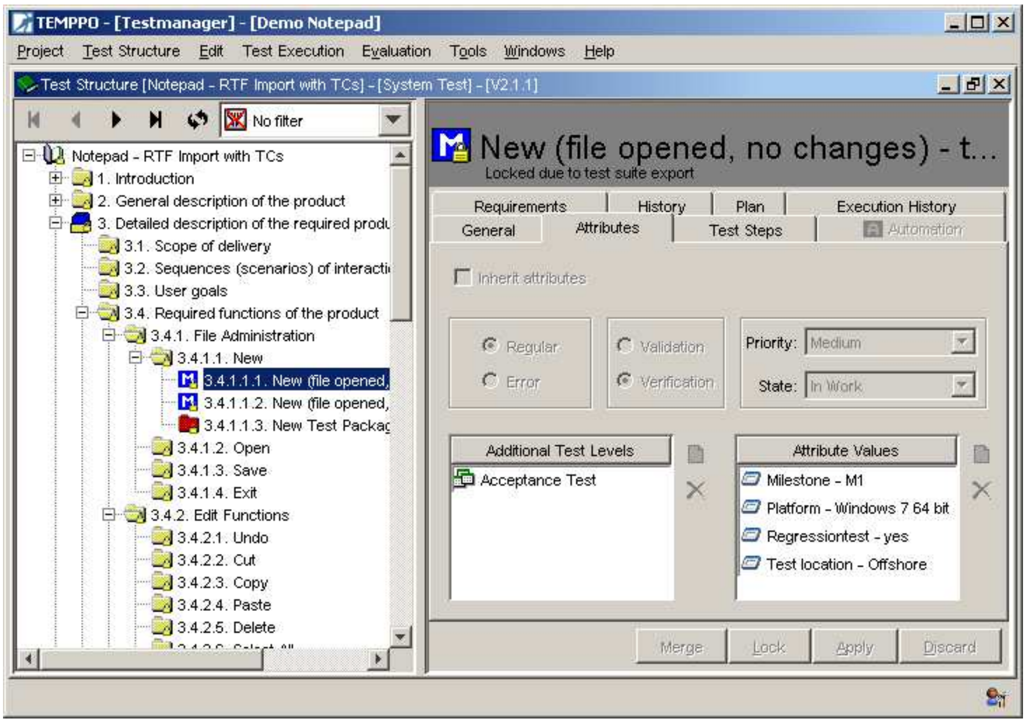
IncludedTPs: Whole test structure

ExcludedTCs:

Figure 108 - Locked test suite

In the tab **Export Info** all exports (of course more than one are possible) of this **locked** test suite are listed. Even the test packages are marked locked, which means that the item is read-only. Test cases also cannot be changed, but can be executed on workstations connected to the central database.

The corresponding test structure will also be locked. Figure 109 shows that also test cases are marked read-only. The only possible change is to add test results.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V2.1.1]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required produ
3.1. Scope of delivery
3.2. Sequences (scenarios) of interacti
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened,
3.4.1.1.2. New (file opened,
3.4.1.1.3. New Test Packag
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.2.1. Undo
3.4.2.2. Cut
3.4.2.3. Copy
3.4.2.4. Paste
3.4.2.5. Delete
New (file opened, no changes) - t...
Locked due to test suite export
Requirements
History
Plan
Execution History
General
Attributes
Test Steps
Automation
Inherit attributes
Regular
Error
Validation
Verification
Priority:
State:
Additional Test Levels
Attribute Values
Merge
Lock
Apply
Discard

☐ Inherit attributes

● Regular ○ Error

○ Validation ● Verification

Priority: Medium

State: In Work

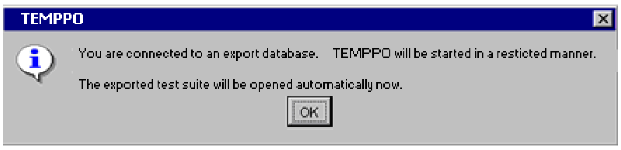
Additional Test Levels: Acceptance Test

Attribute Values:
- Milestone - M1
- Platform - Windows 7 64 bit
- Regressiontest - yes
- Test location - Offshore

Figure 109 - Locked test structure

3.1.4.7.2Test case execution with an exported test suite

Let’s continue the example of offline test case execution: You’ll close TEMPPO and unplug your notebook from the central database (LAN). Unplugged from the central database you first start the TEMPPO Administrator and select the export database. If you start TEMPPO with the export database you are informed that TEMPPO is open in a restricted manner (see Figure 110), which means that only the **Test Execution, Find** and **Evaluation** functionality is available and only the test suite window is opened. Test structures are not displayed in that restricted mode.



You are connected to an export database. TEMPPO will be started in a resticted manner.

The exported test suite will be opened automatically now.

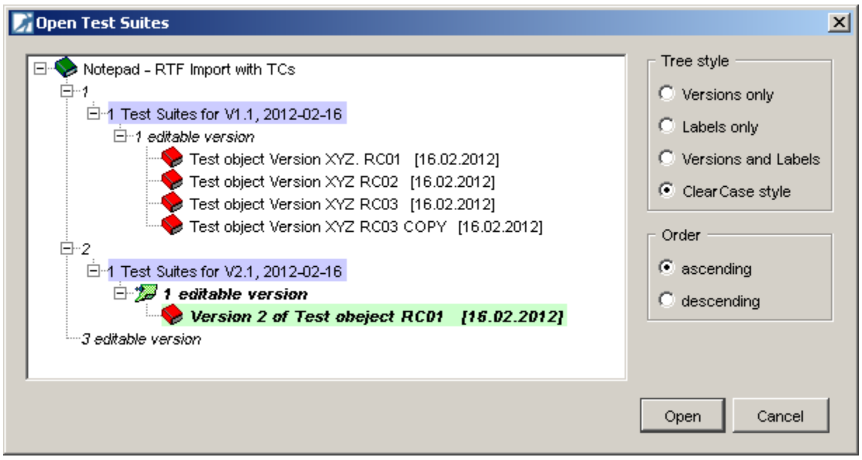
Figure 110 – Warning: TEMPPO in a restricted manner

During test case execution the following values/text fields can be changed:

- Result Output: Result text for the whole test case.
- Actual Result: For each test step
- Bug ID
- Results: Test results for each step (passed, failed, blocked, not executed)

3.1.4.7.3Importing test results

Returning from test case execution the testers want to import the results to the central database. You have to connect to the LAN and select the central database in the TEMPPO Administrator. After opening TEMPPO Test Manager you have to open the corresponding test structure and suite. The (read-only) test suite is marked with a small lock (see Figure 111) and colored blue, if it matches to the test suite in the export database.



Open Test Suites

```
Notepad - RTF Import with TCs
  1
    1 Test Suites for V1.1, 2012-02-16
      1 editable version
        Test object Version XYZ. RC01   [16.02.2012]
        Test object Version XYZ RC02   [16.02.2012]
        Test object Version XYZ RC03   [16.02.2012]
        Test object Version XYZ RC03 COPY   [16.02.2012]
  2
    1 Test Suites for V2.1, 2012-02-16
      1 editable version
        Version 2 of Test obeject RC01   [16.02.2012]
  3 editable version
```

Tree style
○ Versions only
○ Labels only
○ Versions and Labels
● ClearCase style

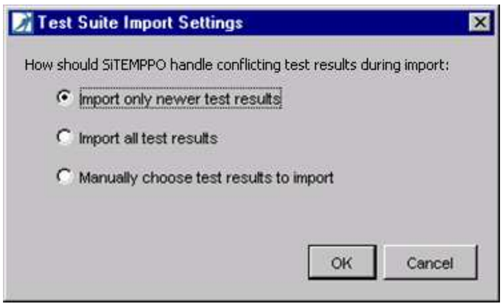
Order
● ascending
○ descending

Figure 111 - Locked test suites

ⓘ **The opened test suite now shows the results of central database and not of your export database.**

For importing the results from the export database, you have to activate the menu item **Import Test Suite**. All tests that were NOT executed for the first time cause no problems. They are imported without any conflicts. TEMPPO Test Manager provides 3 possibilities to import the test results to handle conflicts (see Figure 112).

Then an open-dialog is shown, where the export database has to be selected.



Test Suite Import Settings

How should SiTEMPPO handle conflicting test results during import:

● Import only newer test results
○ Import all test results
○ Manually choose test results to import

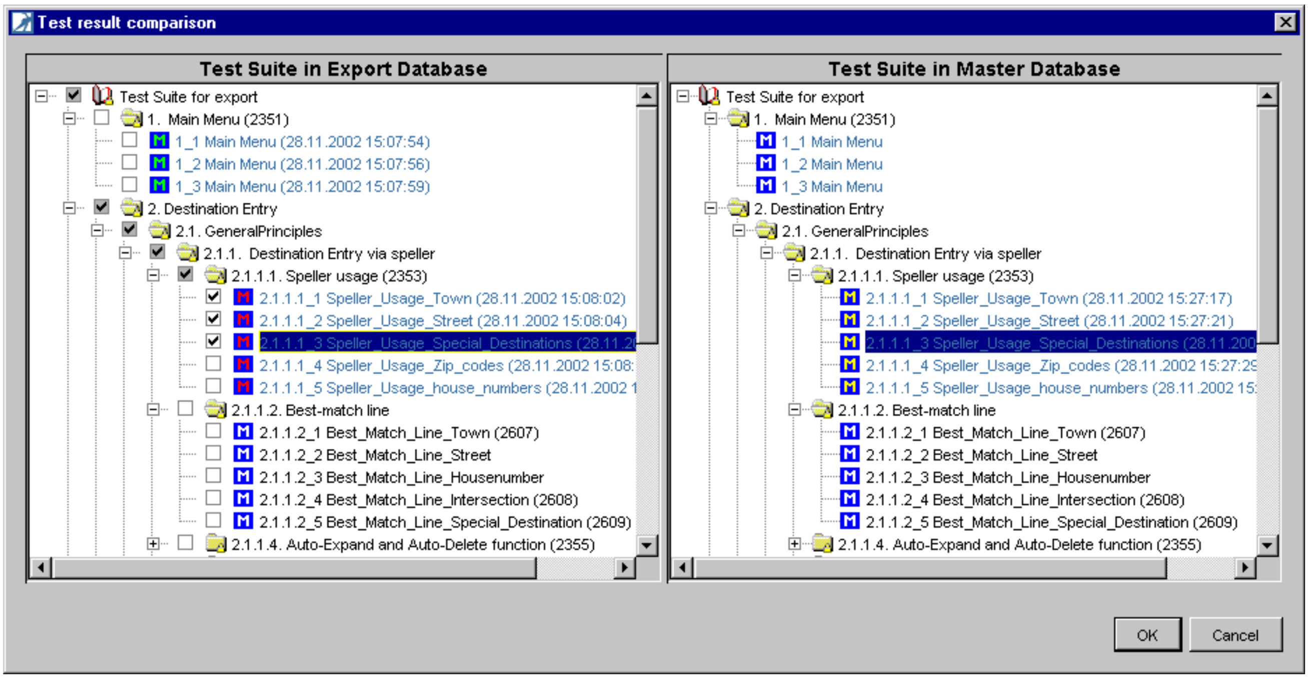
Figure 112 - Test Suite Import Settings

Import only newer test results: If a user already imported a test result for a test case and you also executed the same test case

- Earlier, the result in the central database won't be overridden by yours.
- Later, your result will be written into the database. The other result is lost.
- Import all test results: All test results of already executed test cases are overwritten by your results

Manually choose test results to import: When activating that possibility, Figure 113 - Manual import of test results is shown. In the right part of the window the current (result) state of test cases in the central database is shown together with their execution date (NOT importing date!). Test cases that may cause conflicts are colored blue. The left side of the window shows the test results of the exported test suite. Now you can activate the checkboxes of test cases that you want to import.

ⓘ **Note: All test results that are marked are overwritten in the central database after closing the window with OK.**



Test Suite in Export Database

☑ Test Suite for export
□ 1. Main Menu (2351)
□ 1_1 Main Menu (28.11.2002 15:07:54)
□ 1_2 Main Menu (28.11.2002 15:07:56)
□ 1_3 Main Menu (28.11.2002 15:07:59)
☑ 2. Destination Entry
☑ 2.1. GeneralPrinciples
☑ 2.1.1. Destination Entry via speller
☑ 2.1.1.1. Speller usage (2353)
☑ 2.1.1.1_1 Speller_Usage_Town (28.11.2002 15:08:02)
☑ 2.1.1.1_2 Speller_Usage_Street (28.11.2002 15:08:04)
☑ 2.1.1.1_3 Speller_Usage_Special_Destinations (28.11.2
□ 2.1.1.1_4 Speller_Usage_Zip_codes (28.11.2002 15:08:
□ 2.1.1.1_5 Speller_Usage_house_numbers (28.11.2002 1
□ 2.1.1.2. Best-match line
□ 2.1.1.2_1 Best_Match_Line_Town (2607)
□ 2.1.1.2_2 Best_Match_Line_Street
□ 2.1.1.2_3 Best_Match_Line_Housenumber
□ 2.1.1.2_4 Best_Match_Line_Intersection (2608)
□ 2.1.1.2_5 Best_Match_Line_Special_Destination (2609)
□ 2.1.1.4. Auto-Expand and Auto-Delete function (2355)

Test Suite in Master Database

Test Suite for export
1. Main Menu (2351)
1_1 Main Menu
1_2 Main Menu
1_3 Main Menu
2. Destination Entry
2.1. GeneralPrinciples
2.1.1. Destination Entry via speller
2.1.1.1. Speller usage (2353)
2.1.1.1_1 Speller_Usage_Town (28.11.2002 15:27:17)
2.1.1.1_2 Speller_Usage_Street (28.11.2002 15:27:21)
2.1.1.1_3 Speller_Usage_Special_Destinations (28.11.200
2.1.1.1_4 Speller_Usage_Zip_codes (28.11.2002 15:27:2
2.1.1.1_5 Speller_Usage_house_numbers (28.11.2002 15:
2.1.1.2. Best-match line
2.1.1.2_1 Best_Match_Line_Town (2607)
2.1.1.2_2 Best_Match_Line_Street
2.1.1.2_3 Best_Match_Line_Housenumber
2.1.1.2_4 Best_Match_Line_Intersection (2608)
2.1.1.2_5 Best_Match_Line_Special_Destination (2609)
2.1.1.4. Auto-Expand and Auto-Delete function (2355)

Figure 113 - Manual import of test results

3.1.4.7.4Unlocking the test suite

As a last resort, it is possible to manually unlock a test suite (see chapter 5.5.4).

3.1.5 Evaluation

In this phase test structures and test suites are analyzed and reports are created. You may analyze any version of your test structure (for switching to another test structure version refer to 3.3.10).

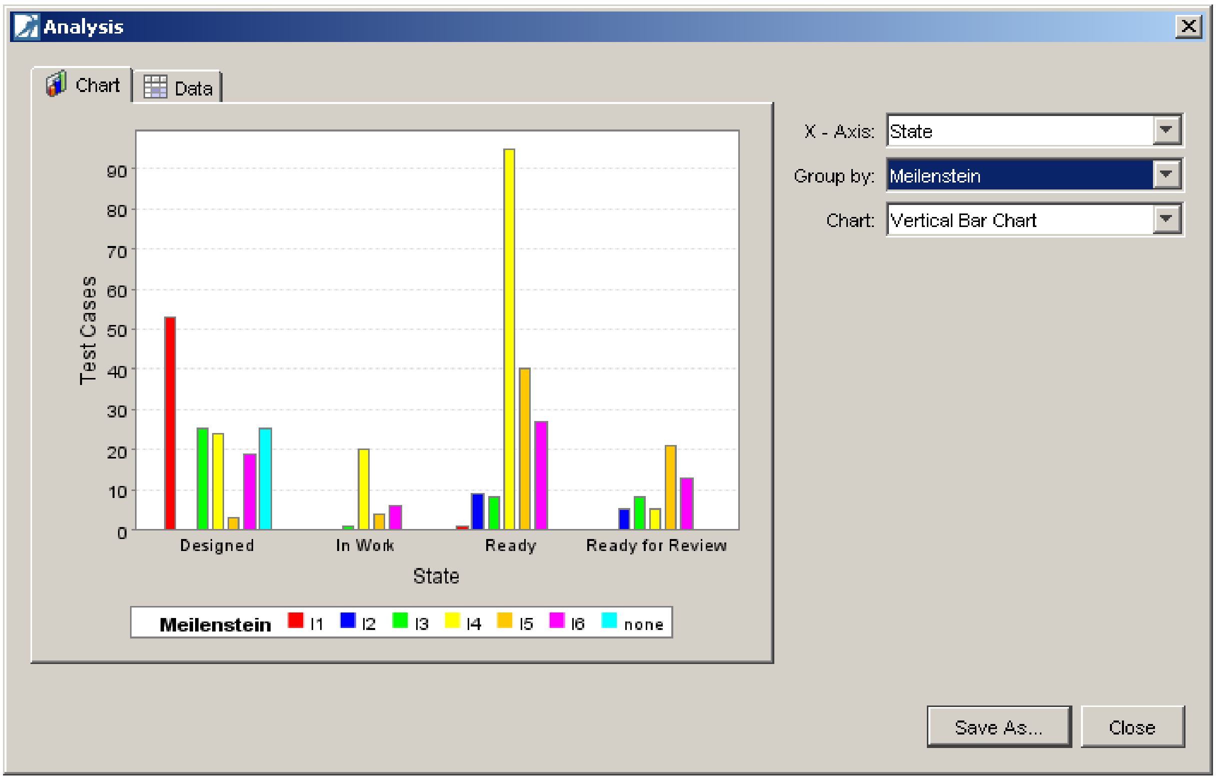
3.1.5.1 Analysis

The analysis feature is used to give you a quick overview about your test structure or test suite. You create an analysis chart by selecting the node to analyze and activating **Evaluation > Analyse selected > with current opened explorer [project | test structure | test suite] > All….** In the Analysis dialog you can set values for X – Axis and group values on the Y - Axis. The following charts for displaying the data are possible:

- Horizontal bar chart
- Horizontal bar chart (3D)
- Line chart
- Stacked horizontal bar chart
- Stacked horizontal bar chart (3D)
- Stacked vertical bar chart
- Stacked vertical bar chart (3D)
- Vertical bar chart
- Vertical bar chart (3D)

There is also the possibility to save the chart as .jpg or .png file by pressing the button `Save As…`

Figure 115 and Figure 115 show an analysis from a test structure and a test suite, displaying the number of test cases for each priority grouped by the state.

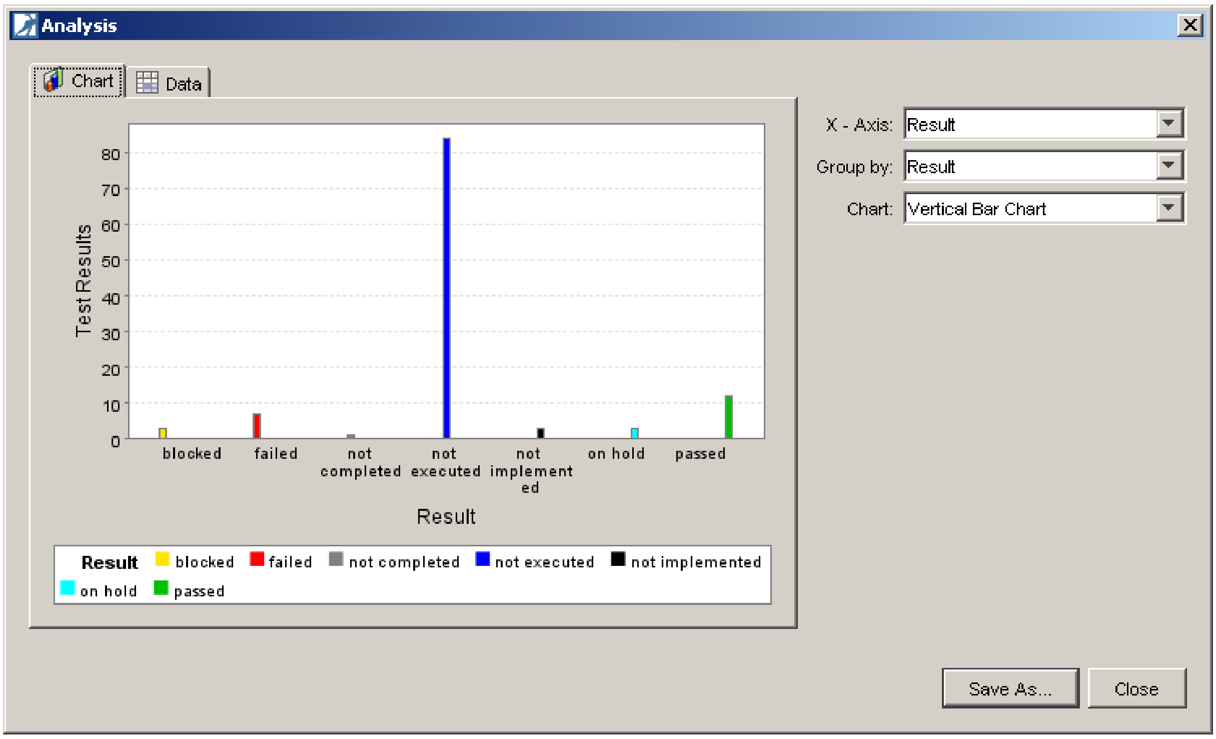


X - Axis: State

Group by: Meilenstein

Chart: Vertical Bar Chart

Figure 114 – Test Structure Analysis



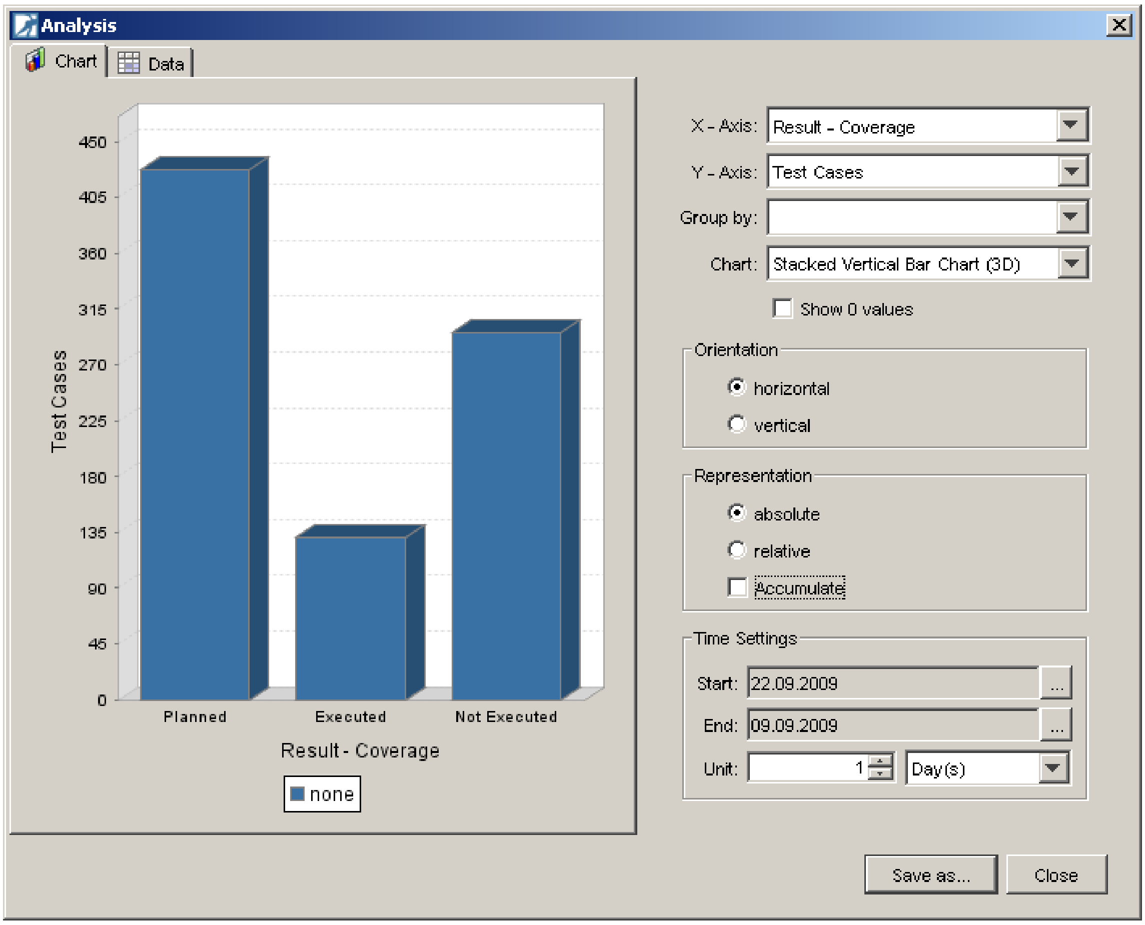
X - Axis: Result

Group by: Result

Chart: Vertical Bar Chart

Figure 115 –Test Suite Analysis

A special analysis is the result coverage: The following analysis compares or displays the planned tests and executed / not executed tests.



X - Axis: Result - Coverage

Y - Axis: Test Cases

Group by:

Chart: Stacked Vertical Bar Chart (3D)

□ Show 0 values

Orientation
● horizontal
○ vertical

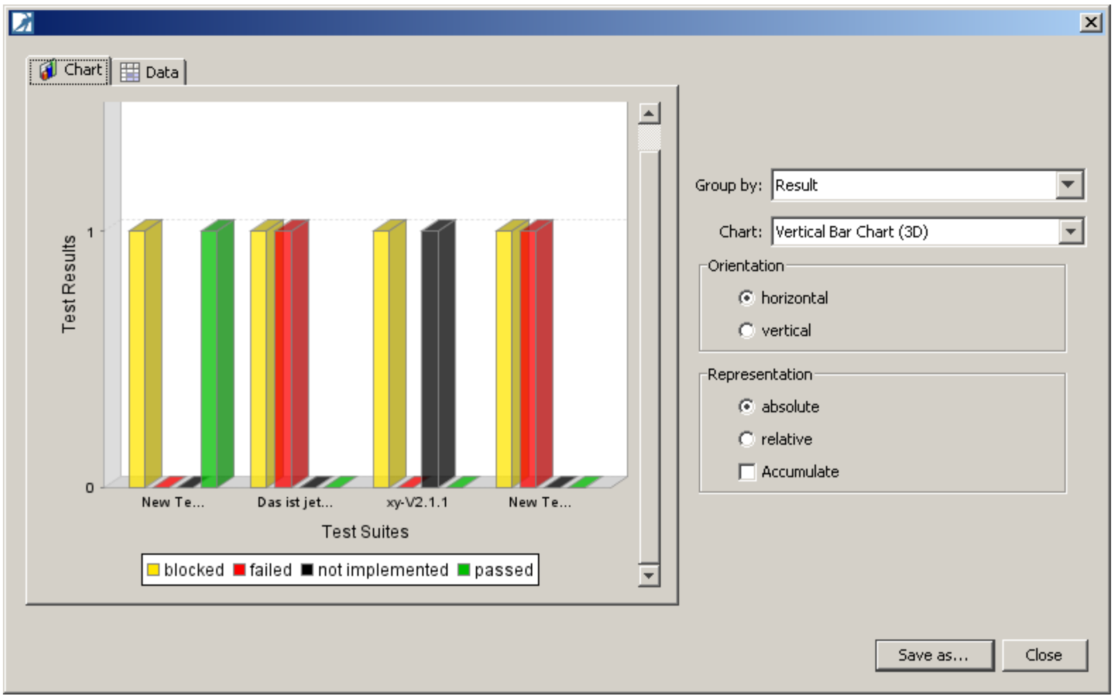
Representation
● absolute
○ relative
□ Accumulate

Time Settings
Start: 22.09.2009
End: 09.09.2009
Unit: 1 Day(s)

Figure 116 - Test Suite Analysis - Coverage

3.1.5.1.1 Analysis with previous test suites

Now there is an analysis through a sequence of test suites (predecessors). Activate **Evaluation > Analyse selected > with current opened explorer > All...** and Figure 117 opens. In the Analysis dialog you can set group values on the Y – Axis and chart type.



Group by: Result

Chart: Vertical Bar Chart (3D)

Orientation
● horizontal
○ vertical

Representation
● absolute
○ relative
□ Accumulate

Figure 117 – Analysis with previous test suites

There is also the possibility to save the chart as jpg or png file by pressing the button **Save As…**

3.1.5.2 Requirement analysis

You create a special analysis chart by selecting the node to analyze and activating **Evaluation > Analyse selected -> Requirements only….**

At first you have to select the requirement structure which you want to analyze.

If the requirement structure is imported from an RM tool, only the selectable requirements are analyzed.

If you want to analyze the requirement attributes, you should select an attribute. Then the analysis is calculated with the values of attribute (see example: Figure 123).

Figure 118 shows a requirement analysis, if the plan value for the coverage is greater than 1 test case. The requirement GEN-FN-0030 is covered by 2 test cases but the plan value was 3. Therefore the requirement is displayed red, means: Coverage = false.

The following chart types for displaying the data are possible:

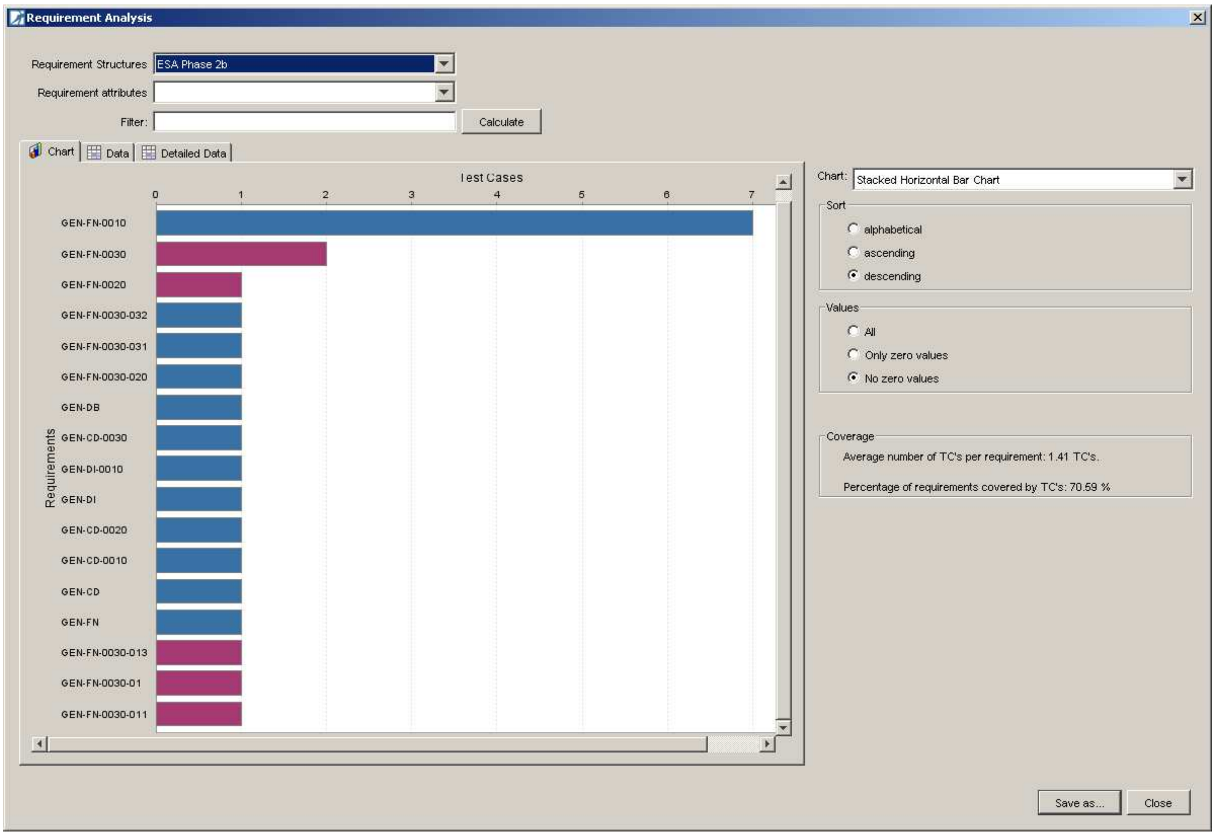
- Horizontal bar chart
- Horizontal bar chart (3D)
- Stacked horizontal bar chart
- Stacked horizontal bar chart (3D)

There is also the possibility to save the chart as jpg or png file by pressing the button `Save As…`

Figure 119, Figure 122, and Figure 121 show an analysis from a requirement structure, displaying the number of assignments from a requirement to test cases. The test cases linked to the requirements are displayed as

- Chart
- Numbers

- Test case IDs



Requirement Analysis

Requirement Structures: ESA Phase 2b

Requirement attributes:

Filter:

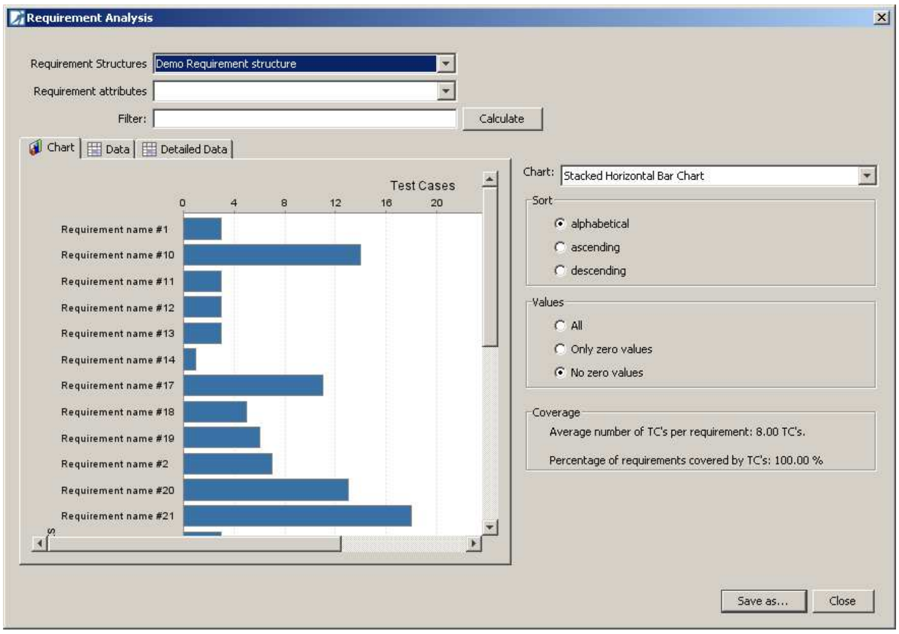
Chart: Stacked Horizontal Bar Chart

Sort
○ alphabetical
○ ascending
● descending

Values
○ All
○ Only zero values
● No zero values

Coverage
Average number of TC's per requirement: 1.41 TC's.
Percentage of requirements covered by TC's: 70.59 %

Figure 118 - Planned coverage



Requirement Analysis

Requirement Structures: Demo Requirement structure

Requirement attributes:

Filter:

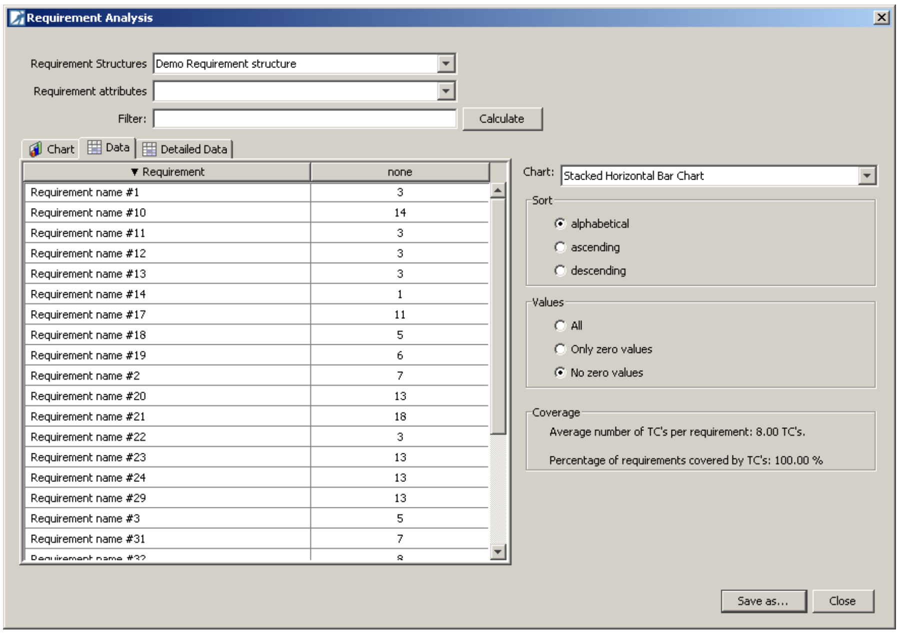
Chart: Stacked Horizontal Bar Chart

Sort
● alphabetical
○ ascending
○ descending

Values
○ All
○ Only zero values
● No zero values

Coverage
Average number of TC's per requirement: 8.00 TC's.
Percentage of requirements covered by TC's: 100.00 %

Figure 119 - Test Structure Requirement Analysis



Requirement Structures: Demo Requirement structure

Requirement attributes:

Filter:

Chart: Stacked Horizontal Bar Chart

Sort
- ● alphabetical
- ○ ascending
- ○ descending

Values
- ○ All
- ○ Only zero values
- ● No zero values

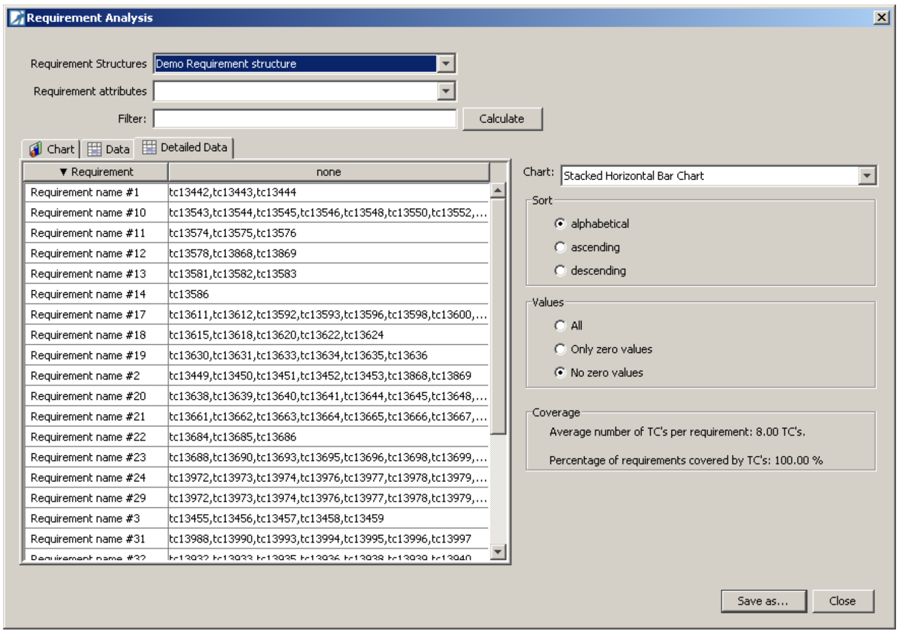
Coverage

Average number of TC's per requirement: 8.00 TC's.

Percentage of requirements covered by TC's: 100.00 %

▼ Requirement	none
Requirement name #1	3
Requirement name #10	14
Requirement name #11	3
Requirement name #12	3
Requirement name #13	3
Requirement name #14	1
Requirement name #17	11
Requirement name #18	5
Requirement name #19	6
Requirement name #2	7
Requirement name #20	13
Requirement name #21	18
Requirement name #22	3
Requirement name #23	13
Requirement name #24	13
Requirement name #29	13
Requirement name #3	5
Requirement name #31	7
Requirement name #32	8

Figure 120 - Test Structure Requirement Analysis - Data



Requirement Structures: Demo Requirement structure

Requirement attributes:

Filter:

Chart: Stacked Horizontal Bar Chart

Sort
- ● alphabetical
- ○ ascending
- ○ descending

Values
- ○ All
- ○ Only zero values
- ● No zero values

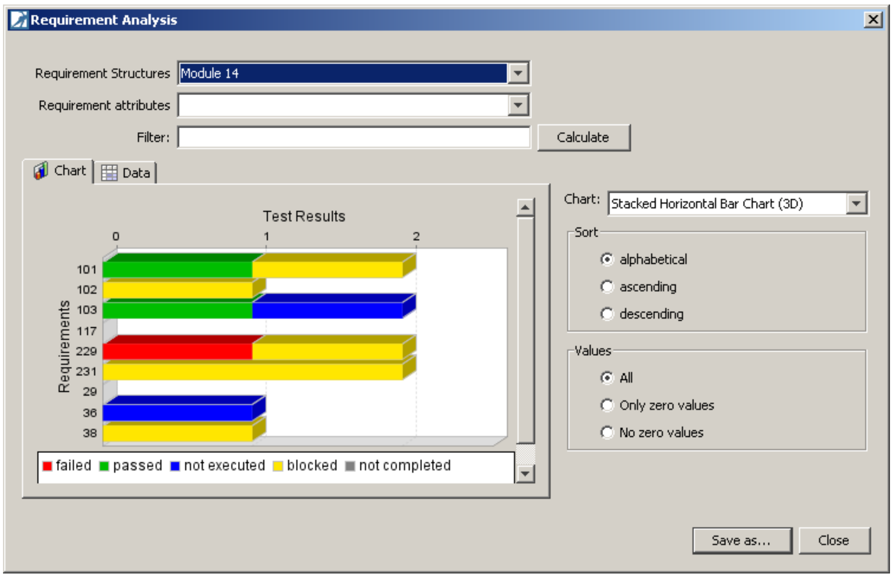
Coverage

Average number of TC's per requirement: 8.00 TC's.

Percentage of requirements covered by TC's: 100.00 %

▼ Requirement	none
Requirement name #1	tc13442,tc13443,tc13444
Requirement name #10	tc13543,tc13544,tc13545,tc13546,tc13548,tc13550,tc13552,...
Requirement name #11	tc13574,tc13575,tc13576
Requirement name #12	tc13578,tc13868,tc13869
Requirement name #13	tc13581,tc13582,tc13583
Requirement name #14	tc13586
Requirement name #17	tc13611,tc13612,tc13592,tc13593,tc13596,tc13598,tc13600,...
Requirement name #18	tc13615,tc13618,tc13620,tc13622,tc13624
Requirement name #19	tc13630,tc13631,tc13633,tc13634,tc13635,tc13636
Requirement name #2	tc13449,tc13450,tc13451,tc13452,tc13453,tc13868,tc13869
Requirement name #20	tc13638,tc13639,tc13640,tc13641,tc13644,tc13645,tc13648,...
Requirement name #21	tc13661,tc13662,tc13663,tc13664,tc13665,tc13666,tc13667,...
Requirement name #22	tc13684,tc13685,tc13686
Requirement name #23	tc13688,tc13690,tc13693,tc13695,tc13696,tc13698,tc13699,...
Requirement name #24	tc13972,tc13973,tc13974,tc13976,tc13977,tc13978,tc13979,...
Requirement name #29	tc13972,tc13973,tc13974,tc13976,tc13977,tc13978,tc13979,...
Requirement name #3	tc13455,tc13456,tc13457,tc13458,tc13459
Requirement name #31	tc13988,tc13990,tc13993,tc13994,tc13995,tc13996,tc13997
Requirement name #32	tc13932,tc13933,tc13935,tc13936,tc13938,tc13939,tc13940

Figure 121 - Test Structure Requirement Analysis - Detailed Data


Requirement Analysis

Requirement Structures: Module 14

Requirement attributes:

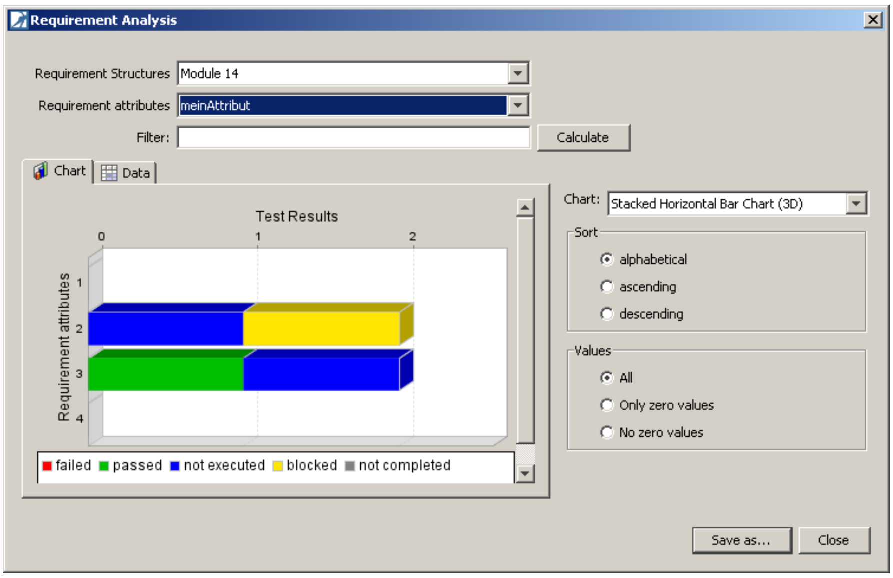
Filter:

Chart: Stacked Horizontal Bar Chart (3D)

Sort: ● alphabetical ○ ascending ○ descending

Values: ● All ○ Only zero values ○ No zero values

Figure 122 - Test Suite Requirement Analysis


Requirement Analysis

Requirement Structures: Module 14

Requirement attributes: meinAttribut

Filter:

Chart: Stacked Horizontal Bar Chart (3D)

Sort: ● alphabetical ○ ascending ○ descending

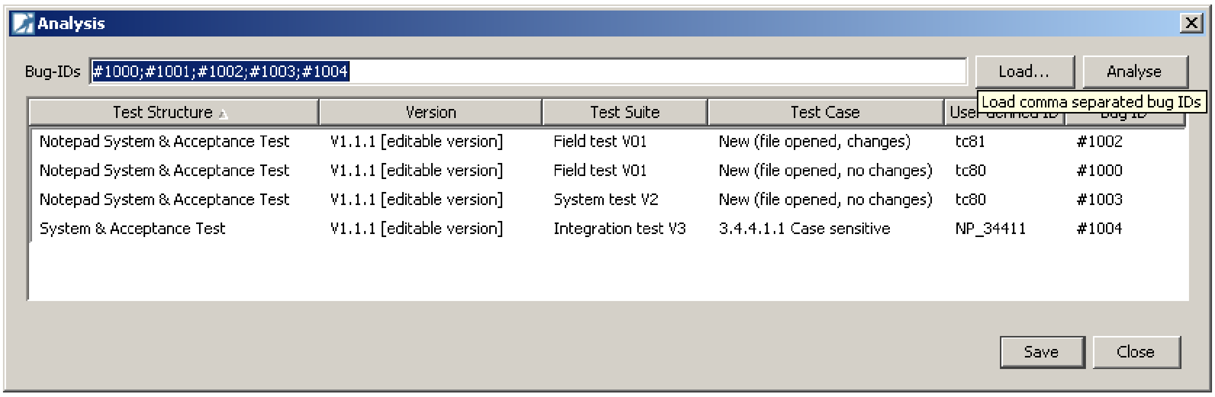
Values: ● All ○ Only zero values ○ No zero values

Figure 123 – Test Suite Requirement Analysis with Attributes

3.1.5.3 Analysis by Bug ID (project)

Consider the situation of a new version of the test object and a list of bugs are fixed in order to retest, but you do not know the test cases or test structures. TEMPPO Test Manager offers the possibility to load or enter a list of **bug IDs** and display the linked information: **Test structure, Version, Test Suite, Test Case, Test Case ID** and **Bug ID**.

If a project is opened the corresponding analysis window can be activated and Figure 125 is displayed. You have the possibility either to load a comma separated file of Bug IDs or enter them directly. After pressing the button **Analyse** the window shows the Bug IDs together with **Test structure, Version, Test Suite, Test Case, Test Case ID**.



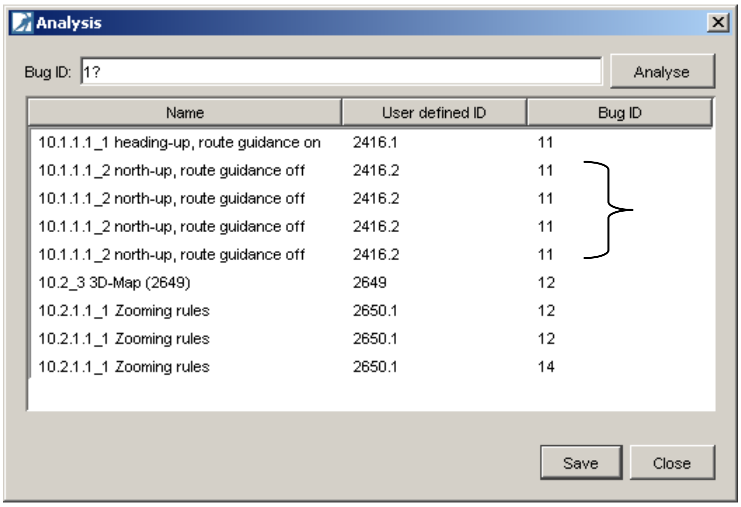
Bug-IDs: #1000;#1001;#1002;#1003;#1004

Test Structure	Version	Test Suite	Test Case	User defined ID	Bug ID
Notepad System & Acceptance Test	V1.1.1 [editable version]	Field test V01	New (file opened, changes)	tc81	#1002
Notepad System & Acceptance Test	V1.1.1 [editable version]	Field test V01	New (file opened, no changes)	tc80	#1000
Notepad System & Acceptance Test	V1.1.1 [editable version]	System test V2	New (file opened, no changes)	tc80	#1003
System & Acceptance Test	V1.1.1 [editable version]	Integration test V3	3.4.4.1.1 Case sensitive	NP_34411	#1004

Figure 124 - Analysis by Bug ID (project)

3.1.5.4 Analysis by Bug ID (test suite)

TEMPPO Test Manager offers you the possibility of analyzing the suite for creating a list of test cases that are referenced by certain bug IDs. By activating the menu item **Analyse selected -> Bug ID**, Figure 125 is displayed.



Bug ID: 1?

Name	User defined ID	Bug ID
10.1.1.1_1 heading-up, route guidance on	2416.1	11
10.1.1.1_2 north-up, route guidance off	2416.2	11
10.1.1.1_2 north-up, route guidance off	2416.2	11
10.1.1.1_2 north-up, route guidance off	2416.2	11
10.1.1.1_2 north-up, route guidance off	2416.2	11
10.2_3 3D-Map (2649)	2649	12
10.2.1.1_1 Zooming rules	2650.1	12
10.2.1.1_1 Zooming rules	2650.1	12
10.2.1.1_1 Zooming rules	2650.1	14

A bug ID like "1?" is reported
in different test steps
of the same test case

Figure 125 - Analysis by Bug ID (test suite)

In the text box **Bug ID** you can enter the bug ID or a regular expression, if you want to create a list of more or all test cases that are referenced. By pressing the button **Analyse**, the test case list is shown and can be saved as csv file by pressing the button **Save**.

By selecting a test case in the window **Analysis**, the cases are also highlighted in the test suite tree.

3.1.5.5 Progress charts

TEMPPO offers a feature, which allows you to create flexible progress charts for test structures and test suites. On one hand you can compare progress of test case creation and on the other hand you can display the progress of test suites by comparing the results.

3.1.5.5.1 Creation Progress

If a test structure is opened and menu item **Creation Progress** is activated, Figure 126 is shown. By pressing the button **Select Test Structures** Figure 127 is displayed, where you can select from the test structure version tree with the mouse together with **<CTRL>** pressed the versions to be displayed.

If you want to display several consecutive versions you have to select the first one, keep holding the **<SHIFT>** key and select the last one in the tree (normal selecting behavior).

The following chart types are available:

- Horizontal bar chart
- Horizontal bar chart (3D)
- Line chart
- Stacked horizontal bar chart
- Stacked horizontal bar chart (3D)
- Stacked vertical bar chart
- Stacked vertical bar chart (3D)
- Vertical bar chart
- Vertical bar chart (3D)

With the radio button **Orientation** you can display the description of the x-axis either **vertically** or **horizontally**.

The data can be switched from **absolute** to **relative** and vice versa.

Using the checkboxes **States** the chart can be customized.

By pressing the button **Save As…** the chart can be saved as jpg or png file.

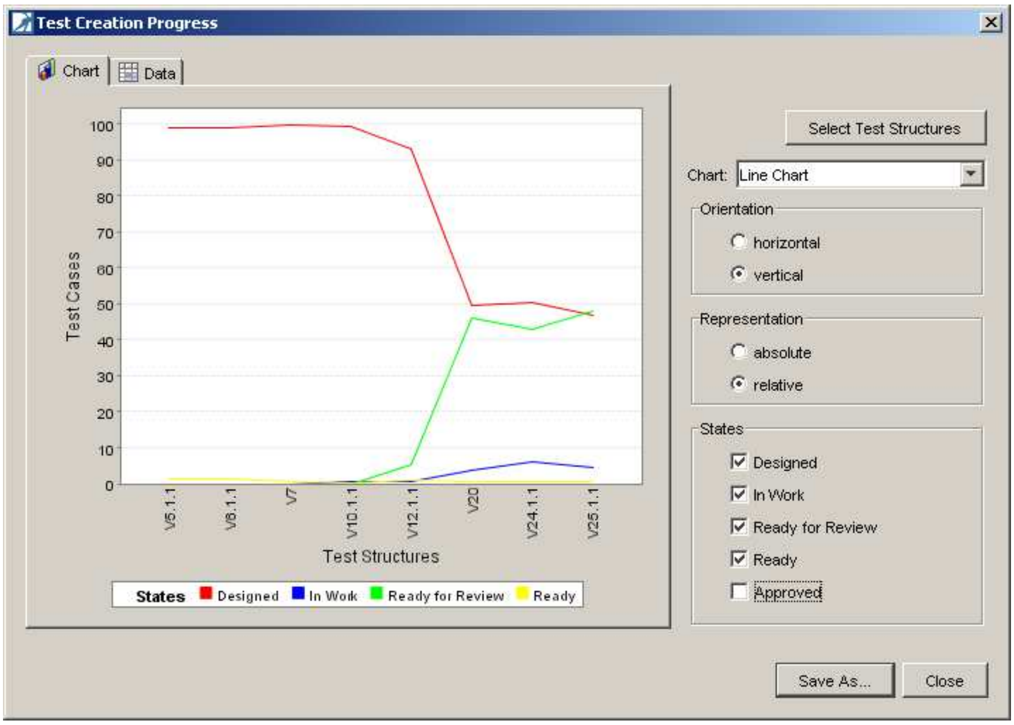


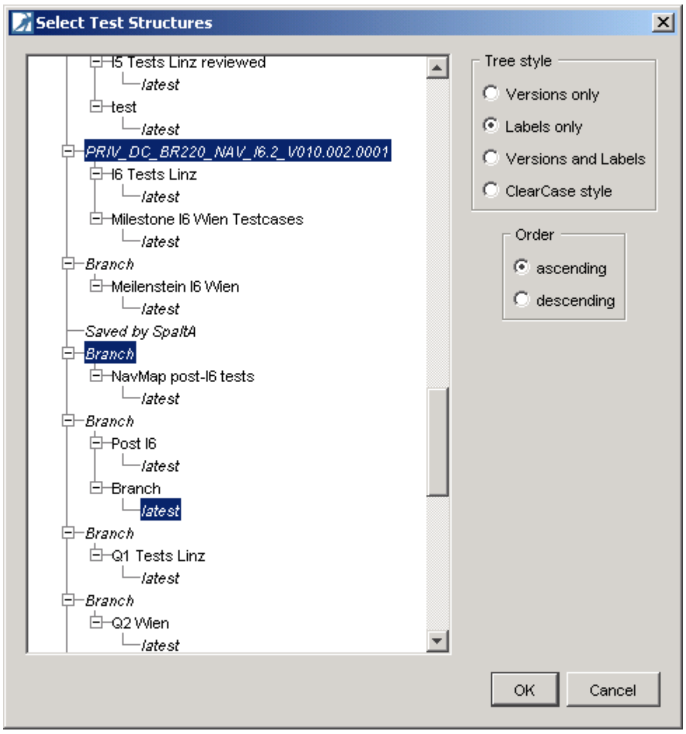
Chart: Line Chart

Orientation
○ horizontal
● vertical

Representation
○ absolute
● relative

States
☑ Designed
☑ In Work
☑ Ready for Review
☑ Ready
☐ Approved

Figure 126 - Test Creation Progress



```
      ⊟-I5 Tests Linz reviewed
          └latest
      ⊟-test
          └latest
  ⊟-PRIV_DC_BR220_NAV_I6.2_V010.002.0001
      ⊟-I6 Tests Linz
          └latest
      ⊟-Milestone I6 Wien Testcases
          └latest
  ⊟-Branch
      ⊟-Meilenstein I6 Wien
          └latest
  ─Saved by SpaltA
  ⊟-Branch
      ⊟-NavMap post-I6 tests
          └latest
  ⊟-Branch
      ⊟-Post I6
          └latest
      ⊟-Branch
          └latest
  ⊟-Branch
      ⊟-Q1 Tests Linz
          └latest
  ⊟-Branch
      ⊟-Q2 Wien
          └latest
```

Tree style
○ Versions only
● Labels only
○ Versions and Labels
○ ClearCase style

Order
● ascending
○ descending

Figure 127 - Select Test Structures

3.1.5.5.2 Execution Progress

There are 3 analysis available:

- **Standard:** Execution progress of test suites can be created. For several test suites which can be selected results and planned values a chart can be created.

- **Filtered:** Additionally, a filter can be set for execution progress which is applied on all selected test suites. This analysis is slower because all data have to be loaded from database for filtering.
- **Requirements only:** A requirement analysis of several suites over several requirement structures can be created. All data will cumulated to one chart.

3.1.5.5.2.1 Standard

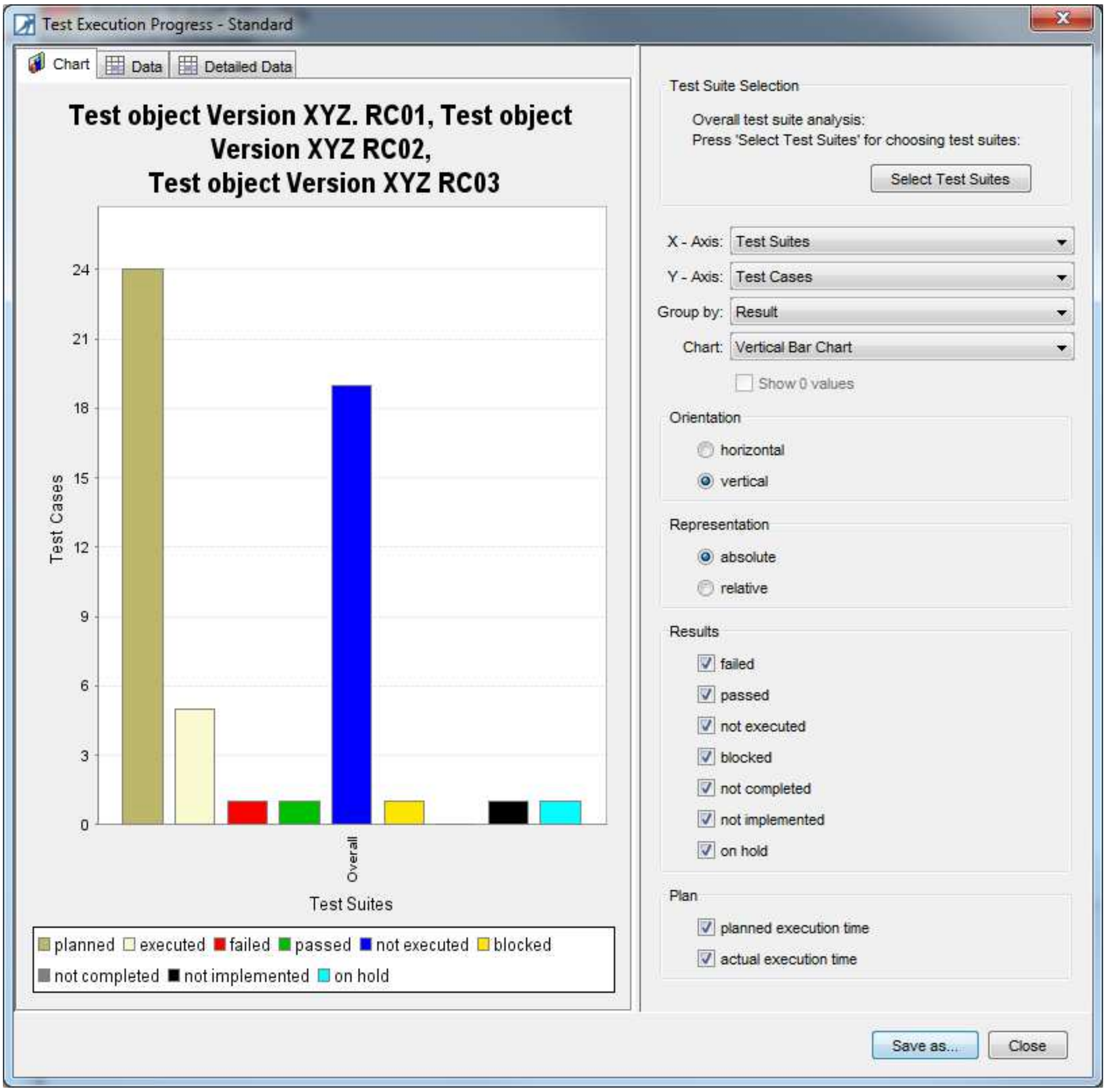
You create an analysis chart of several test suites by activating **Evaluation > Progress charts -> Execution -> Standard…**, if at least a test structure is opened. It is only possible to create a test suites chart of a single test structure.

When opening that analysis, the chart is empty and you can select the test suites in a window that opens if you press the button **Select Test Suites**.

Possible settings:

- **X-axis: Tester, Test suites**
- **Y-axis: Test cases, Planned execution time**
- **Group by: Result**

If **Test Suites** is selected for x-axis, the checkbox **Show 0 values** is greyed, because it makes no sense. Additionally you can (un)display all obsolete results and plan values. The following example - Figure 128– shows the accumulated values of 3 test suites. All 3 test suites contain 24 in sum, therefore 24 test cases were "planned". 5 test cases were executed and 19 test cases were not executed.



Test Suite Selection

Overall test suite analysis:
Press 'Select Test Suites' for choosing test suites:

Select Test Suites

X - Axis: Test Suites

Y - Axis: Test Cases

Group by: Result

Chart: Vertical Bar Chart

☐ Show 0 values

Orientation

○ horizontal
● vertical

Representation

● absolute
○ relative

Results

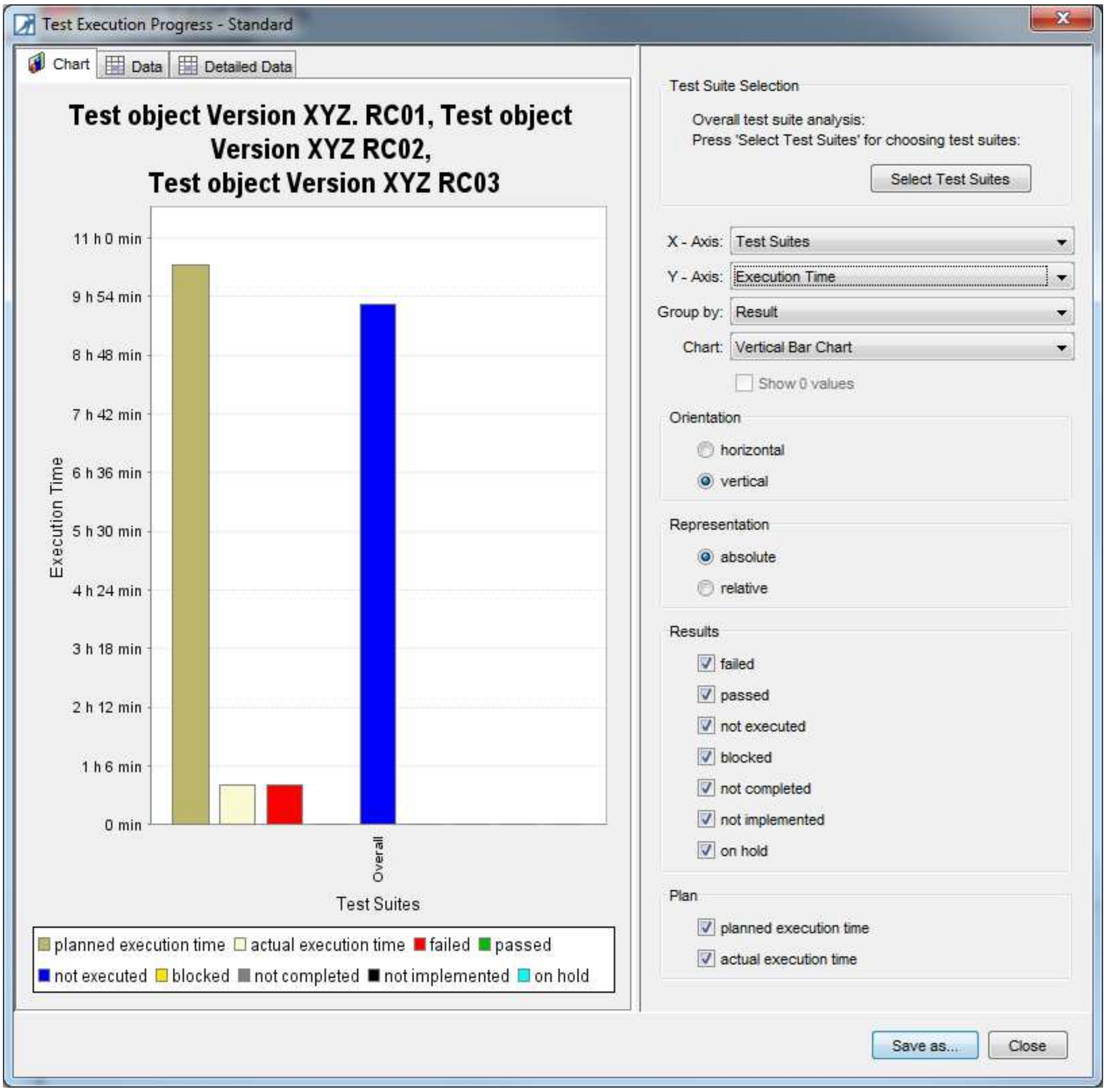
☑ failed
☑ passed
☑ not executed
☑ blocked
☑ not completed
☑ not implemented
☑ on hold

Plan

☑ planned execution time
☑ actual execution time

Figure 128 - Execution progress of several test suites – chart example 1

In the following example **Test Suites** is selected for x-axis and **Execution Time**. You can see that the “Planned execution time” is much higher than the actual one (sum of passed and failed). A test manager can also observe a large amount of remaining test time (not executed ones).



Test Execution Progress - Standard

Chart | Data | Detailed Data

Test object Version XYZ. RC01, Test object Version XYZ RC02, Test object Version XYZ RC03
11 h 0 min
9 h 54 min
8 h 48 min
7 h 42 min
6 h 36 min
5 h 30 min
4 h 24 min
3 h 18 min
2 h 12 min
1 h 6 min
0 min
Execution Time
Overall
Test Suites
▨ planned execution time □ actual execution time 🟥 failed 🟩 passed
🟦 not executed 🟨 blocked ▨ not completed ■ not implemented □ on hold

Test Suite Selection

Overall test suite analysis:
Press 'Select Test Suites' for choosing test suites:

Select Test Suites

X - Axis: Test Suites
Y - Axis: Execution Time
Group by: Result
Chart: Vertical Bar Chart
□ Show 0 values

Orientation
○ horizontal
● vertical

Representation
● absolute
○ relative

Results
☑ failed
☑ passed
☑ not executed
☑ blocked
☑ not completed
☑ not implemented
☑ on hold

Plan
☑ planned execution time
☑ actual execution time

Save as... | Close

Figure 129 - Execution progress of several test suites – chart example 2

3.1.5.5.2.1.1 Detailed data

Tab **Detailed data** shows the data table which can be exported to csv.

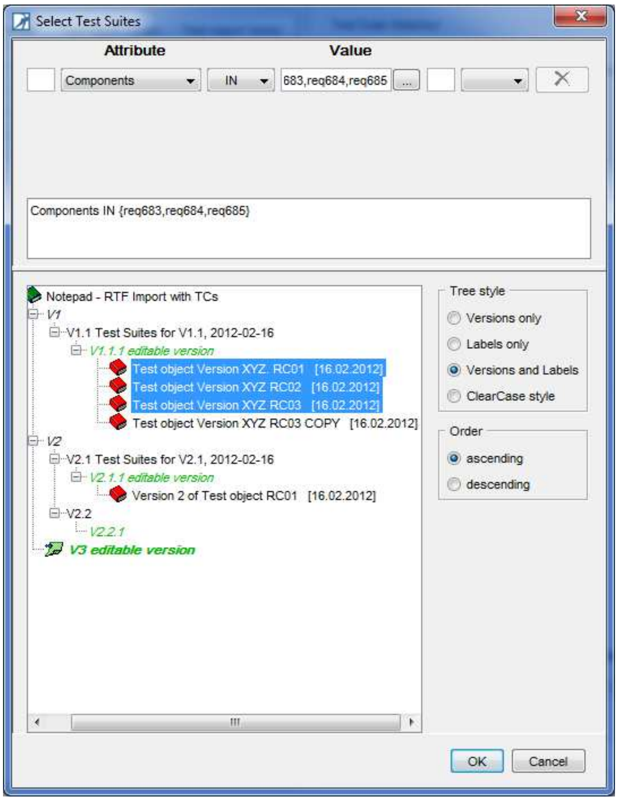
ⓘ **Not all combinations display valid values**

Input values			**Output values**	
x-axis	**y-axis**	**Group by**	**Action x-axis**	**Action y-axis**
Test Suite	Test Case	-	Test case IDs	Result per test suite
Test Suite	Execution Time	-	-	-
Tester	Execution Time	-	-	-
Tester	Test Case	-	Test case IDs	Result per tester

3.1.5.5.2.2 Filtered

If a test suite is opened and menu item **Execution Progress > Filtered...** is activated, Figure 131 is shown.

By pressing the button **Select Test Suites** Figure 130 is displayed, where you can set a filter and select from the test suite tree with the mouse together with <**CTRL**> pressed the suites to be displayed.



Attribute		Value
Components	IN	683,req684,req685

Components IN {req683,req684,req685}

```
Notepad - RTF Import with TCs
V1
  V1.1 Test Suites for V1.1, 2012-02-16
    V1.1.1 editable version
      Test object Version XYZ. RC01   [16.02.2012]
      Test object Version XYZ RC02   [16.02.2012]
      Test object Version XYZ RC03   [16.02.2012]
      Test object Version XYZ RC03 COPY   [16.02.2012]
V2
  V2.1 Test Suites for V2.1, 2012-02-16
    V2.1.1 editable version
      Version 2 of Test object RC01   [16.02.2012]
  V2.2
    V2.2.1
V3 editable version
```

Tree style
○ Versions only
○ Labels only
● Versions and Labels
○ ClearCase style

Order
● ascending
○ descending

Figure 130 – Select Test Suites

If you want to display several consecutive versions you have to select the first one, keep holding the <**SHIFT**> key and select the last one in the tree (normal selecting behavior).

The following chart types are available:

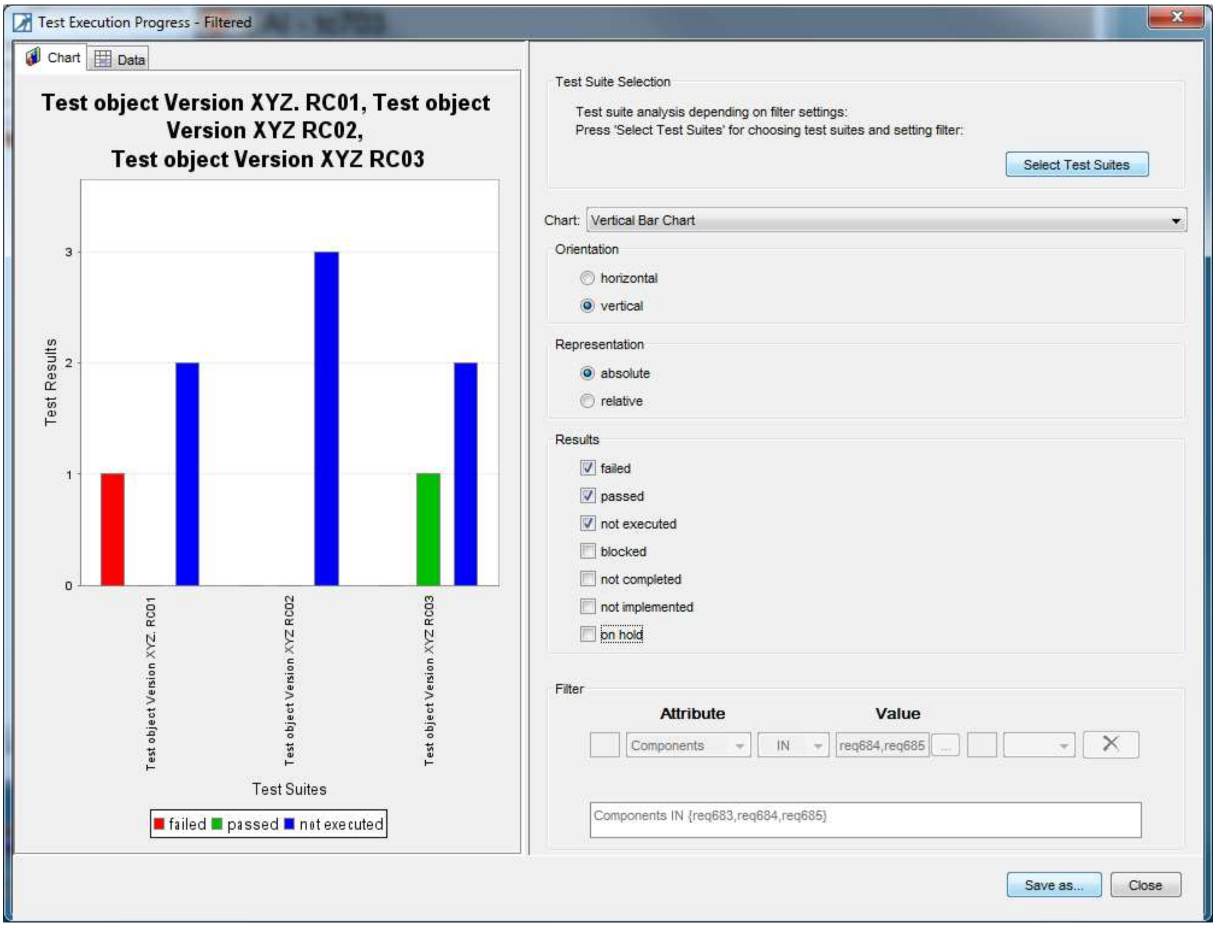
- Horizontal bar chart
- Horizontal bar chart (3D)
- Line chart
- Stacked horizontal bar chart
- Stacked horizontal bar chart (3D)
- Stacked vertical bar chart
- Stacked vertical bar chart (3D)
- Vertical bar chart
- Vertical bar chart (3D)

With the radio button **Orientation** you can display the description of the x-axis either **vertically** or **horizontally**.

The data can be switched from **absolute** to **relative** and vice versa.

Using the checkboxes **Results** the chart can be customized.

By pressing the button **Save As...** the chart can be saved as jpg or png file.


Test Execution Progress - Filtered
Chart
Data
Test object Version XYZ. RC01, Test object Version XYZ RC02, Test object Version XYZ RC03
Test Results
0
1
2
3
Test object Version XYZ. RC01
Test object Version XYZ RC02
Test object Version XYZ RC03
Test Suites
🟥 failed 🟩 passed 🟦 not executed
Select Test Suites
Save as...
Close

Test Suite Selection

Test suite analysis depending on filter settings:
Press 'Select Test Suites' for choosing test suites and setting filter:

Chart: Vertical Bar Chart

Orientation
○ horizontal
● vertical

Representation
● absolute
○ relative

Results
☑ failed
☑ passed
☑ not executed
☐ blocked
☐ not completed
☐ not implemented
☐ on hold

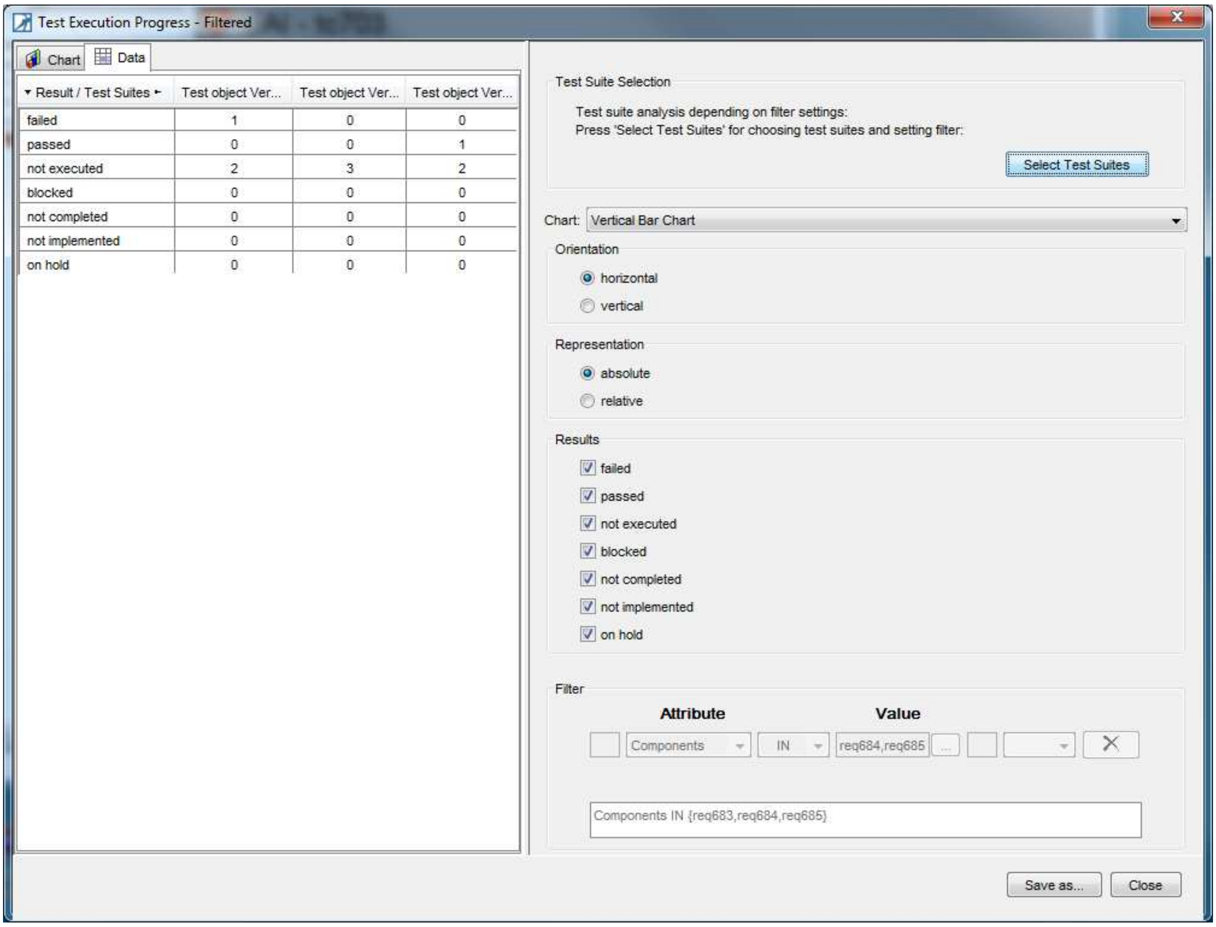
Filter

Attribute		Value
Components	IN	req684,req685

Components IN {req683,req684,req685}

Figure 131 – Text Execution Progress Filtered - Chart

If you activate the tab **Data**, the values are displayed as table and be saved as CSV file.



Test Execution Progress - Filtered

Chart | Data

Result / Test Suites	Test object Ver...	Test object Ver...	Test object Ver...
failed	1	0	0
passed	0	0	1
not executed	2	3	2
blocked	0	0	0
not completed	0	0	0
not implemented	0	0	0
on hold	0	0	0

Test Suite Selection

Test suite analysis depending on filter settings:
Press 'Select Test Suites' for choosing test suites and setting filter:

Select Test Suites

Chart: Vertical Bar Chart

Orientation
- ● horizontal
- ○ vertical

Representation
- ● absolute
- ○ relative

Results
- ☑ failed
- ☑ passed
- ☑ not executed
- ☑ blocked
- ☑ not completed
- ☑ not implemented
- ☑ on hold

Filter

Attribute		Value
Components	IN	req684,req685

Components IN {req683,req684,req685}

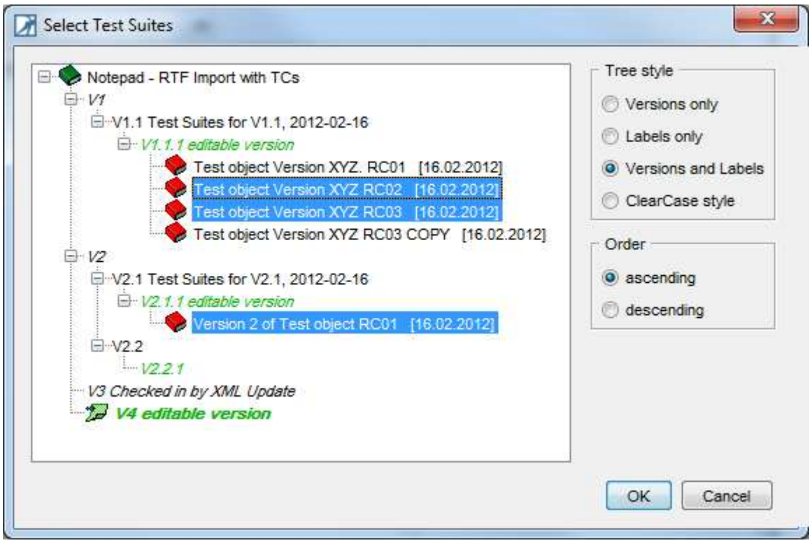
Save as... | Close

Figure 132 – Test Execution Progress Filtered – Data

3.1.5.5.2.3 Requirements only

You create a requirement analysis chart of several test suites by activating **Evaluation > Progress charts -> Execution -> Requirements only...**, if at least a test structure is opened. It is only possible to create a test suites chart of a single test structure.

When opening that analysis, the chart is empty and you can select the test suites in a window that opens if you press the button **Select Test Suites**.



Select Test Suites

```
Notepad - RTF Import with TCs
  V1
    V1.1 Test Suites for V1.1, 2012-02-16
      V1.1.1 editable version
        Test object Version XYZ. RC01   [16.02.2012]
        Test object Version XYZ RC02   [16.02.2012]
        Test object Version XYZ RC03   [16.02.2012]
        Test object Version XYZ RC03 COPY   [16.02.2012]
  V2
    V2.1 Test Suites for V2.1, 2012-02-16
      V2.1.1 editable version
        Version 2 of Test object RC01   [16.02.2012]
    V2.2
      V2.2.1
  V3 Checked in by XML Update
  V4 editable version
```

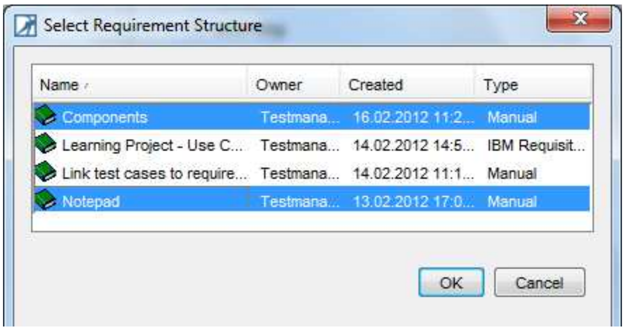
Tree style
- ○ Versions only
- ○ Labels only
- ● Versions and Labels
- ○ ClearCase style

Order
- ● ascending
- ○ descending

OK | Cancel

Figure 133 - Select test suites

Additionally you have to select the requirement structures by pressing the button **Select Structures**.

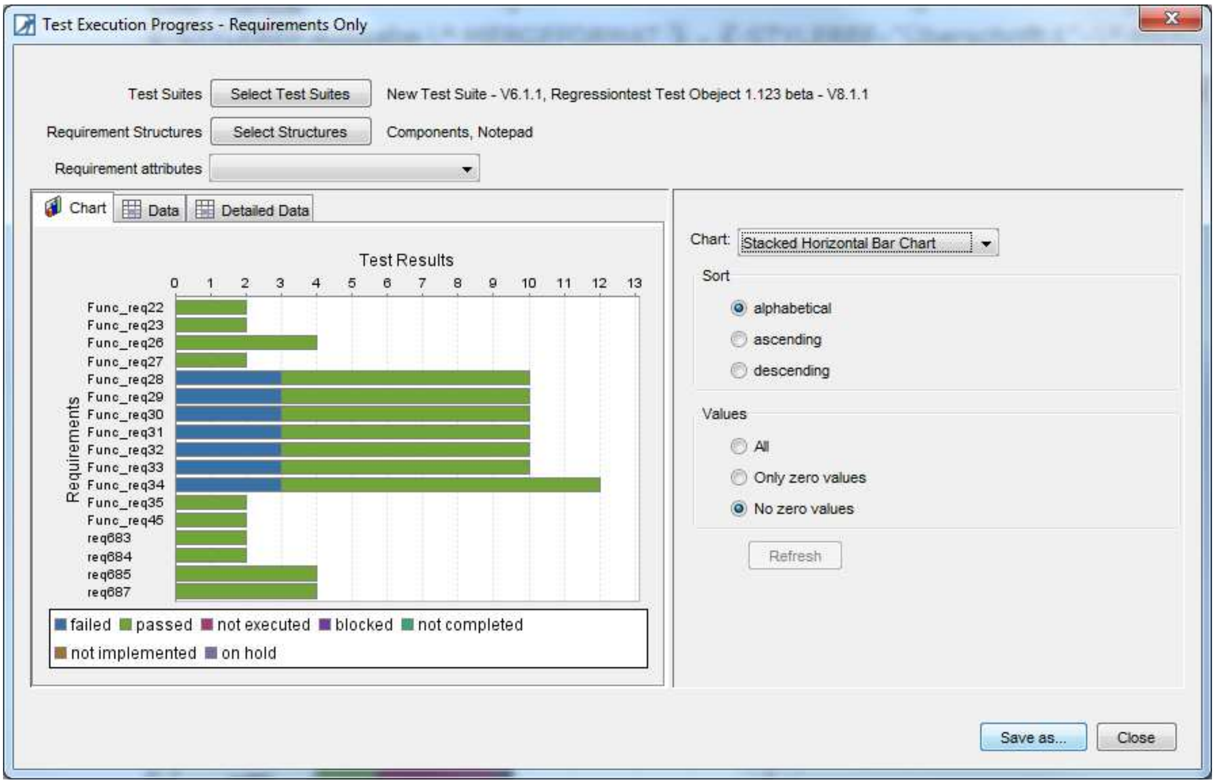


Select Requirement Structure

Name	Owner	Created	Type
Components	Testmana...	16.02.2012 11:2...	Manual
Learning Project - Use C...	Testmana...	14.02.2012 14:5...	IBM Requisit...
Link test cases to require...	Testmana...	14.02.2012 11:1...	Manual
Notepad	Testmana...	13.02.2012 17:0...	Manual

Figure 134 - Select requirement structures

Each time you change anything on test suites or requirement structure selection button **Refresh** get enabled. If you press button **Refresh** chart is refreshed.



Test Execution Progress - Requirements Only

Test Suites: New Test Suite - V6.1.1, Regressiontest Test Obeject 1.123 beta - V8.1.1

Requirement Structures: Components, Notepad

Requirement attributes:

Chart: Stacked Horizontal Bar Chart

Sort: ● alphabetical ○ ascending ○ descending

Values: ○ All ○ Only zero values ● No zero values

Figure 135 - Requirement analysis over several test suites: Chart

Tab chart:

The following chart types are available:

- Horizontal bar chart
- Horizontal bar chart (3D)
- Stacked horizontal bar chart
- Stacked horizontal bar chart (3D)

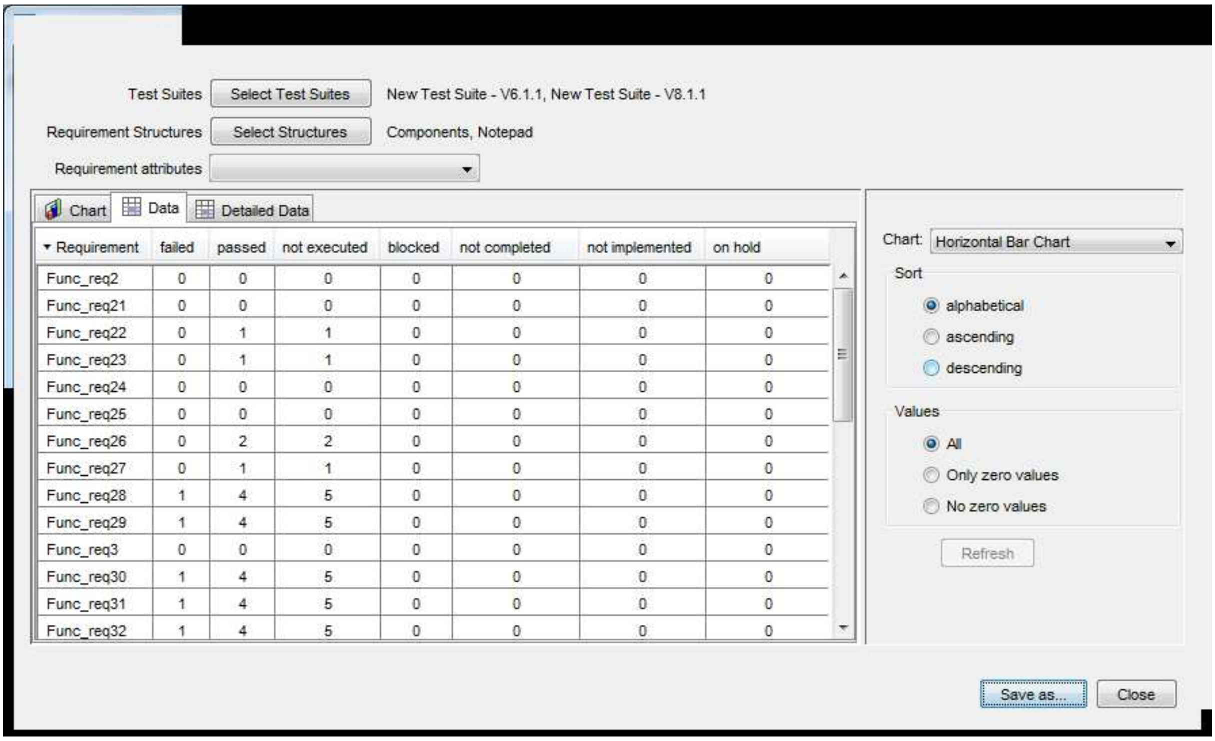
All test cases of the selected test suites with links to the selected requirement structures are cumulated together. This means that one same test case is counted as often as it occurs in the test suites.

On the other all requirements without any link to test cases are displayed.

By pressing the button **Save As…** the chart can be saved as jpg or png file.

Tab Data:

In tab Data number of test results are displayed as table and exported to csv.



Test Suites: New Test Suite - V6.1.1, New Test Suite - V8.1.1

Requirement Structures: Components, Notepad

Requirement attributes:

▾ Requirement	failed	passed	not executed	blocked	not completed	not implemented	on hold
Func_req2	0	0	0	0	0	0	0
Func_req21	0	0	0	0	0	0	0
Func_req22	0	1	1	0	0	0	0
Func_req23	0	1	1	0	0	0	0
Func_req24	0	0	0	0	0	0	0
Func_req25	0	0	0	0	0	0	0
Func_req26	0	2	2	0	0	0	0
Func_req27	0	1	1	0	0	0	0
Func_req28	1	4	5	0	0	0	0
Func_req29	1	4	5	0	0	0	0
Func_req3	0	0	0	0	0	0	0
Func_req30	1	4	5	0	0	0	0
Func_req31	1	4	5	0	0	0	0
Func_req32	1	4	5	0	0	0	0

Chart: Horizontal Bar Chart

Sort
● alphabetical
○ ascending
○ descending

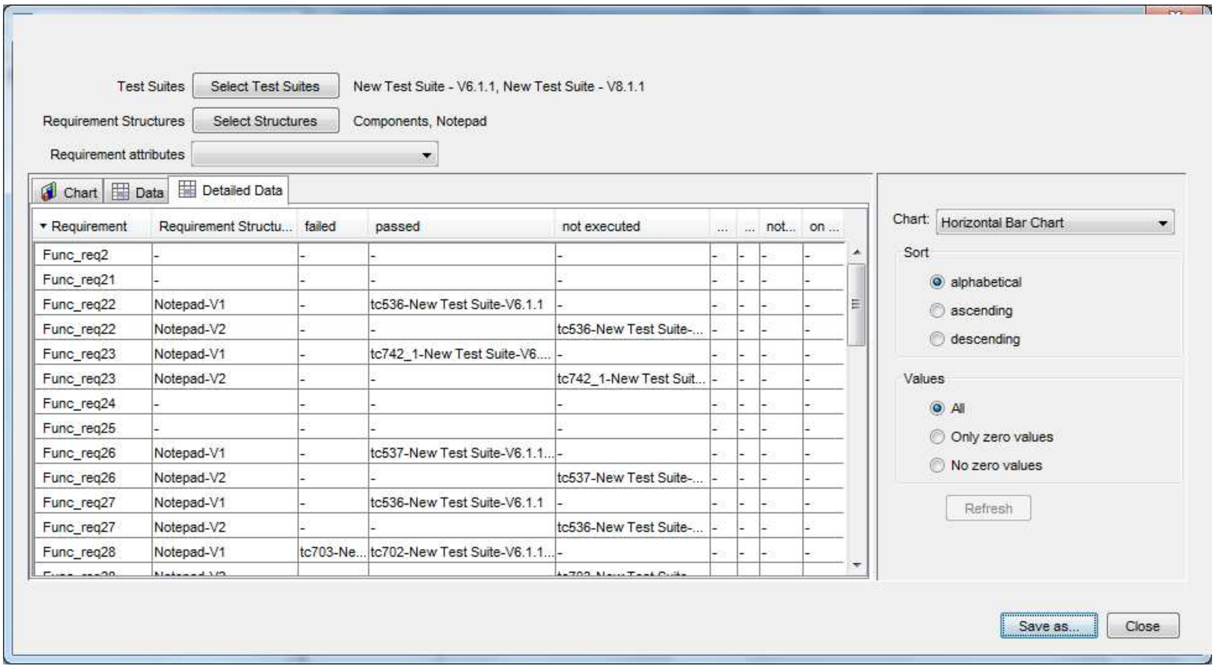
Values
● All
○ Only zero values
○ No zero values

Figure 136 - Requirement analysis over several test suites: Data

In tab **Detailed Data** for each requirement a column **Requirement Structure** and columns for test results with the TC IDs are displayed. For each result cell a list of comma separated values is shown as follows:

<test case ID>-<test suite name>-<test structure version>,

<next test case ID>-<test suite name>-<test structure version>,...



Test Suites: New Test Suite - V6.1.1, New Test Suite - V8.1.1

Requirement Structures: Components, Notepad

Requirement attributes:

Requirement	Requirement Structu...	failed	passed	not executed	...	...	not...	on ...
Func_req2	-	-	-	-	-	-	-	-
Func_req21	-	-	-	-	-	-	-	-
Func_req22	Notepad-V1	-	tc536-New Test Suite-V6.1.1	-	-	-	-	-
Func_req22	Notepad-V2	-	-	tc536-New Test Suite-...	-	-	-	-
Func_req23	Notepad-V1	-	tc742_1-New Test Suite-V6....	-	-	-	-	-
Func_req23	Notepad-V2	-	-	tc742_1-New Test Suit...	-	-	-	-
Func_req24	-	-	-	-	-	-	-	-
Func_req25	-	-	-	-	-	-	-	-
Func_req26	Notepad-V1	-	tc537-New Test Suite-V6.1.1...	-	-	-	-	-
Func_req26	Notepad-V2	-	-	tc537-New Test Suite-...	-	-	-	-
Func_req27	Notepad-V1	-	tc536-New Test Suite-V6.1.1	-	-	-	-	-
Func_req27	Notepad-V2	-	-	tc536-New Test Suite-...	-	-	-	-
Func_req28	Notepad-V1	tc703-Ne...	tc702-New Test Suite-V6.1.1...	-	-	-	-	-

Chart: Horizontal Bar Chart

Sort: ● alphabetical ○ ascending ○ descending

Values: ● All ○ Only zero values ○ No zero values

Figure 137 - Requirement analysis over several test suites: Detailed Data

3.1.5.6 Reporting

TEMPPO Test Manager offers a reporting feature, which allows you to create flexible textual and graphical reports from your test structures and test suites. A test structure or test suite report can be created from any node within the structure, even from a single test case.

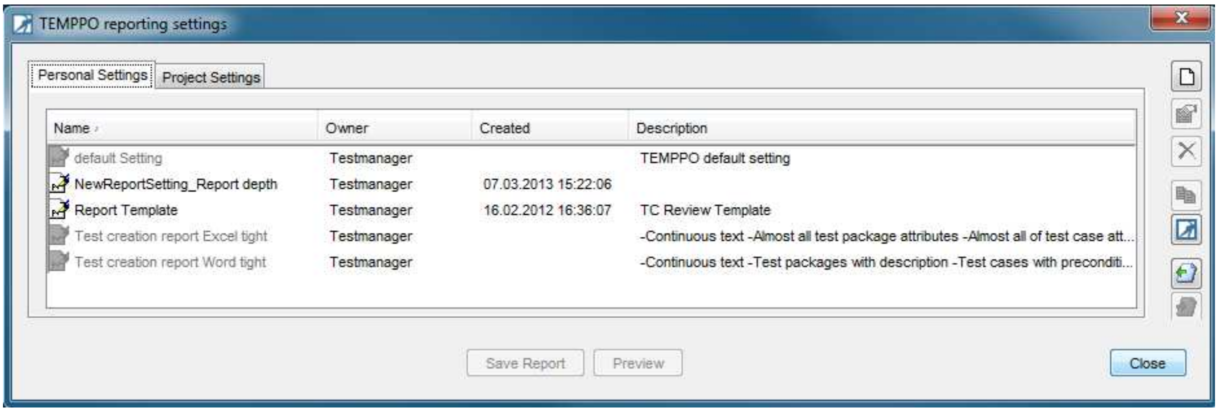
Reports on current select node (root, TP, TC) in a test structure or test suite can be opened via the context menu **Report selected**.

Reports of several test structures versions or test suites are called by the menu item **Evaluation -> Progress charts**.

The following reports are possible:

- If a test structure is opened, reporting for test structure is opened via **Evaluation > Report selected > Opened Test Structure**.
- If a test structure is opened, reporting for more test suites is opened via **Evaluation > Report selected > Several Test Suites**.
- If a test suite is opened, reporting for test suite is opened via **Evaluation > Report selected > Opened Test Suite**.
- If a test structure is opened, reporting for more test suites is opened via **Evaluation > Report selected > Several Test Suites**.

Different report settings can be defined for user level and for projects. In tab **Personal settings**, the user sees its own defined settings and in tab **Project settings**, all settings defined for this project are visible.



Name	Owner	Created	Description
default Setting	Testmanager		TEMPPO default setting
NewReportSetting_Report depth	Testmanager	07.03.2013 15:22:06	
Report Template	Testmanager	16.02.2012 16:36:07	TC Review Template
Test creation report Excel tight	Testmanager		-Continuous text -Almost all test package attributes -Almost all of test case att...
Test creation report Word tight	Testmanager		-Continuous text -Test packages with description -Test cases with preconditi...

Figure 138 – Report settings

With the buttons on the right side, you can add, change and remove settings:

New / Edit
see 3.1.5.6.1

Delete
Setting(s) are deleted.

Copy to other settings
Copy the selected setting(s) to other settings.
Example: Select a setting in tab personal setting and press button. In tab project setting, the copied setting is added.

Import from another project
see 3.1.5.7

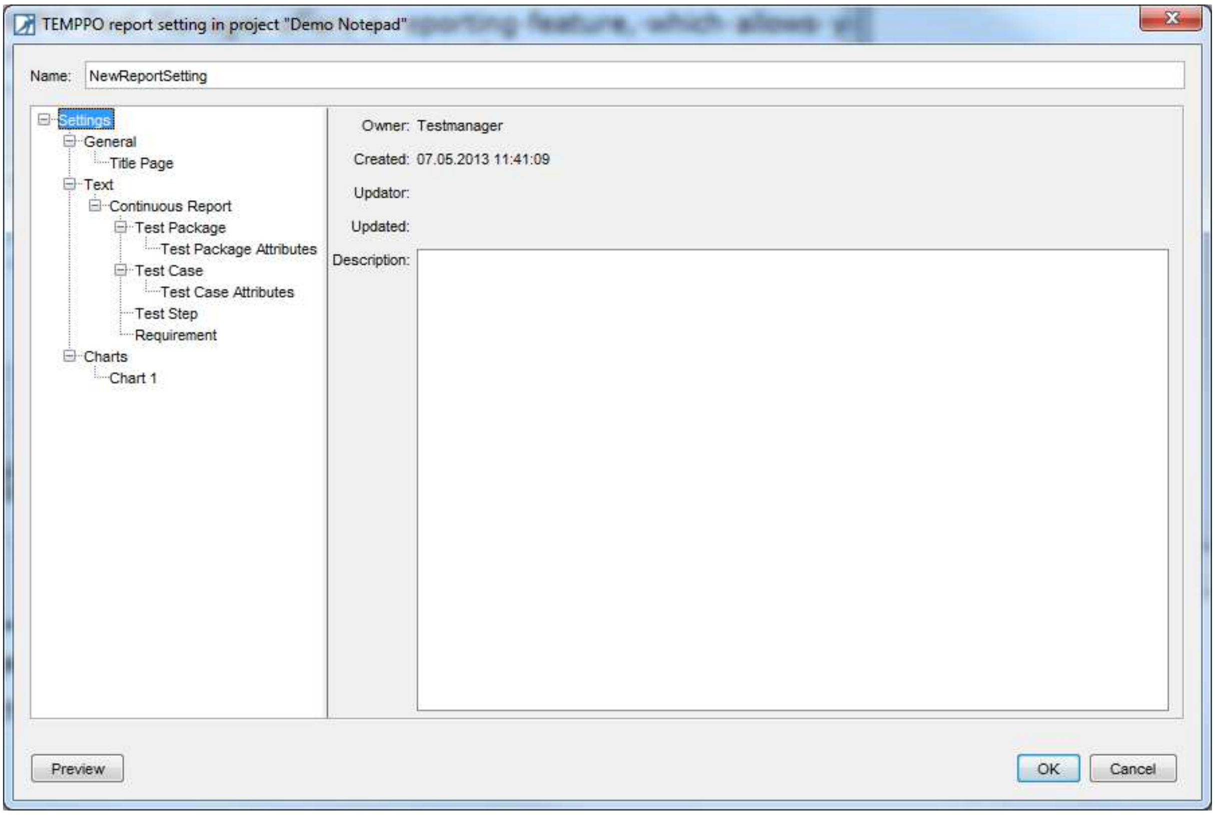
Export to XML file
see 3.1.5.8

Import from XML or TRP-file
see 3.1.5.9

With the buttons below, you can save the report to directory or create a preview. Pressing **Save Report**, a file dialog opens where the directory has to be defined and then report is created.

3.1.5.6.1New / Edit

When pressing the button New or Edit a new window opens, for set the properties for creating a report (see Figure 139).



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

Owner: Testmanager

Created: 07.05.2013 11:41:09

Updator:

Updated:

Description:

Figure 139 – Report settings overview

The reporting functionality is divided into four main setting areas, where each of them again is subdivided into several parts. The areas are described in the following chapters.

3.1.5.6.1.1 General

In the general settings area you define the rough layout and contents of your report.

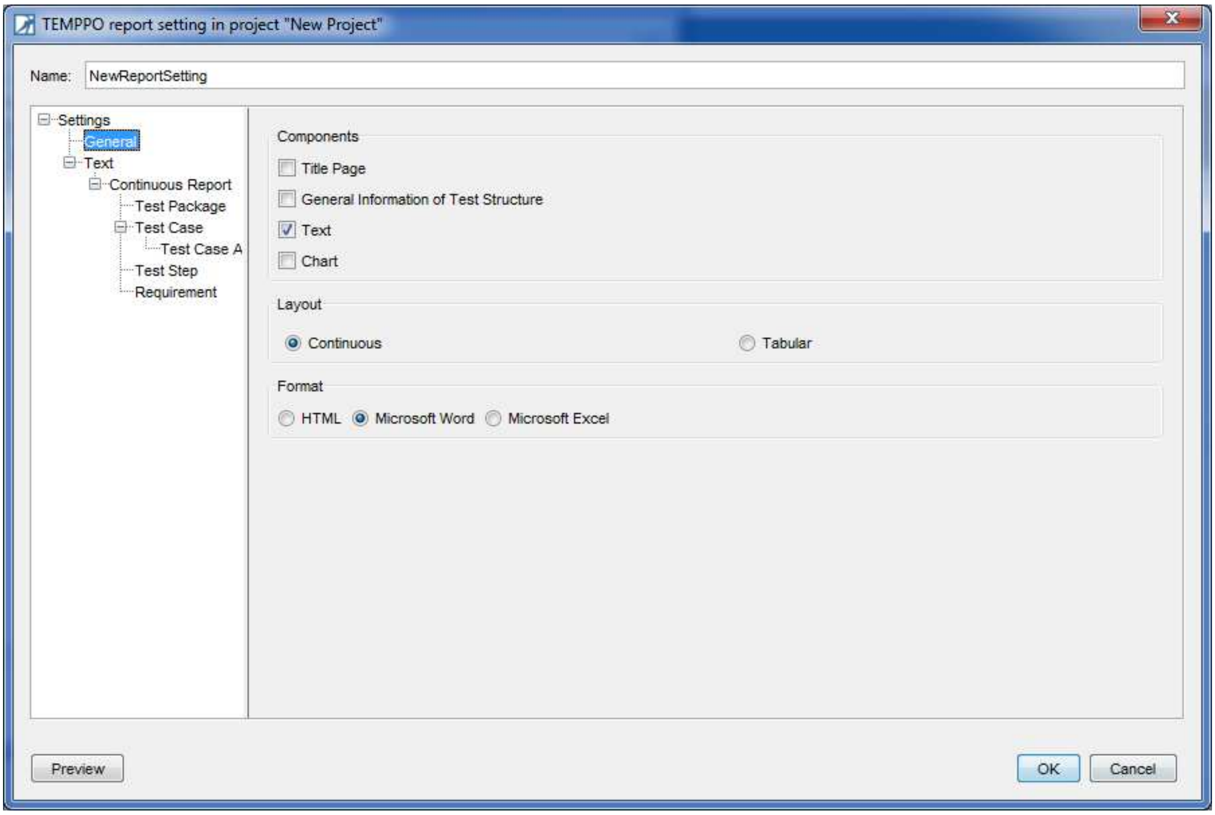
Components

Here you choose the components your report shall contain. Dependent on your selection the tree is changed for doing the relevant settings. The following components are available:

- **Title Page**
- **General Information about Test Structure**
- **Text**
- **Charts**
- **Progress Chart**

Additionally you have to select the **layout**:
Continuous or **Tabular**

and the **Format** for the report:
HTML, Microsoft Word or **Microsoft Excel**



TEMPPO report setting in project "New Project"

Name: NewReportSetting

Components

□ Title Page

□ General Information of Test Structure

☑ Text

□ Chart

Layout

● Continuous ○ Tabular

Format

○ HTML ● Microsoft Word ○ Microsoft Excel

Figure 140 – Reporting: General

If Microsoft Excel is selected, then you can define a header and a footer within the settings.

If **HTML** is selected, reports know from prior TEMPPO versions (5.6 or lower) can be saved.

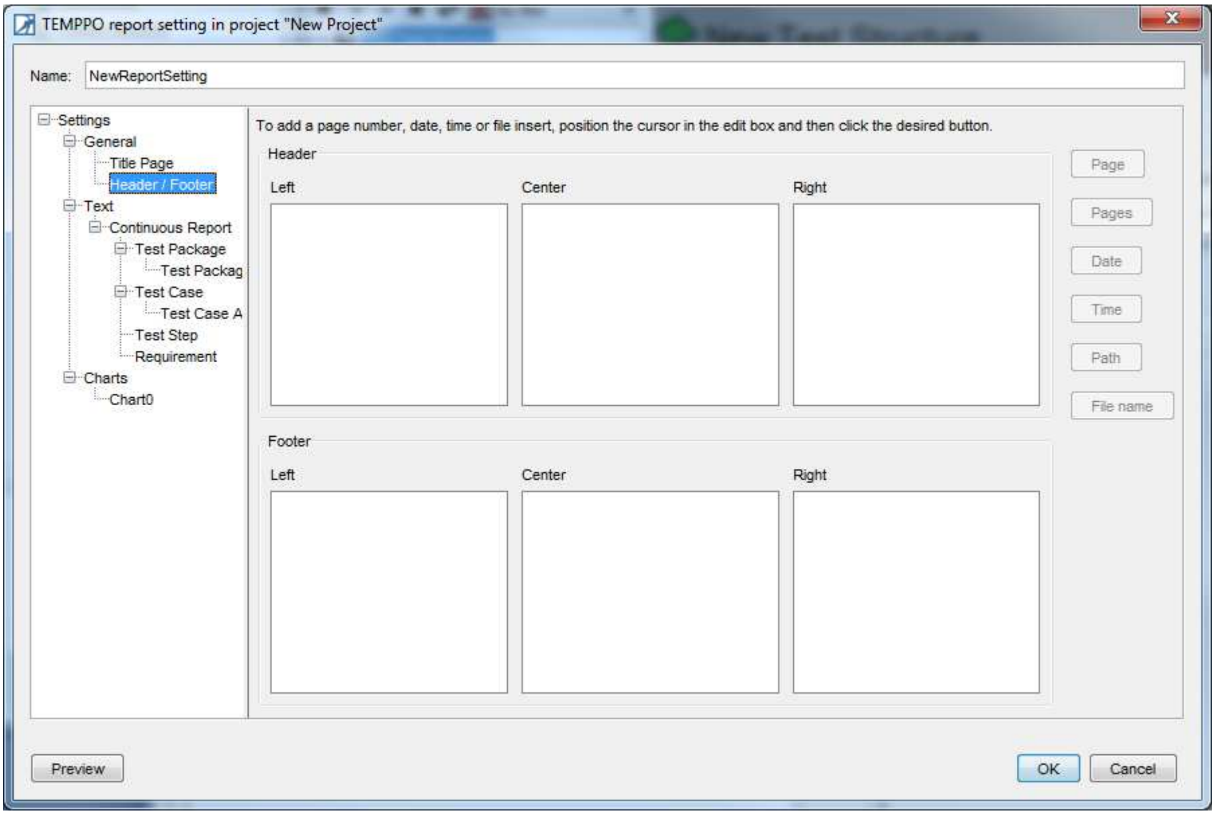
ⓘ **Please, consider that Microsoft Word limits the width of pages to 22 inches (http://support.microsoft.com/?scid=kb%3Ben-us%3B95109&x=14&y=10)**

Title Page

You can customize the title page of your report here, which can contain a title, project name, status, the name of the author and the current date.

Header / Footer

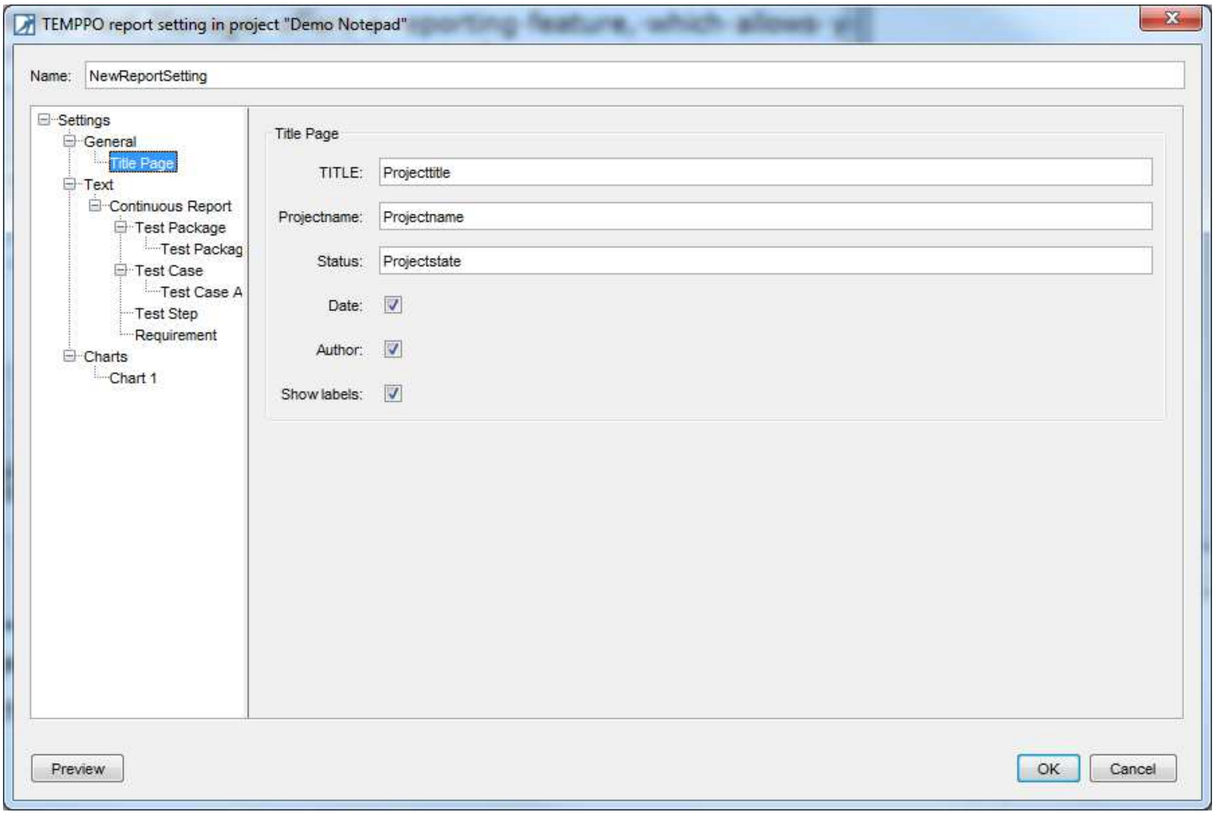
You can customize the header and footer for you report. There are buttons for using the defined keys for page number, date, time etc.



TEMPPO report setting in project "New Project"

Name: NewReportSetting

To add a page number, date, time or file insert, position the cursor in the edit box and then click the desired button.



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

Title Page

TITLE: Projecttitle

Projectname: Projectname

Status: Projectstate

Date: ☑

Author: ☑

Show labels: ☑

Figure 141 – Report: General Settings

3.1.5.6.1.2 Text

Text settings are relevant for the textual data contained in your report and how they are displayed.

General Text Settings:

First you can choose the **Layout**, if you want your data to be displayed in continuous text or table format.

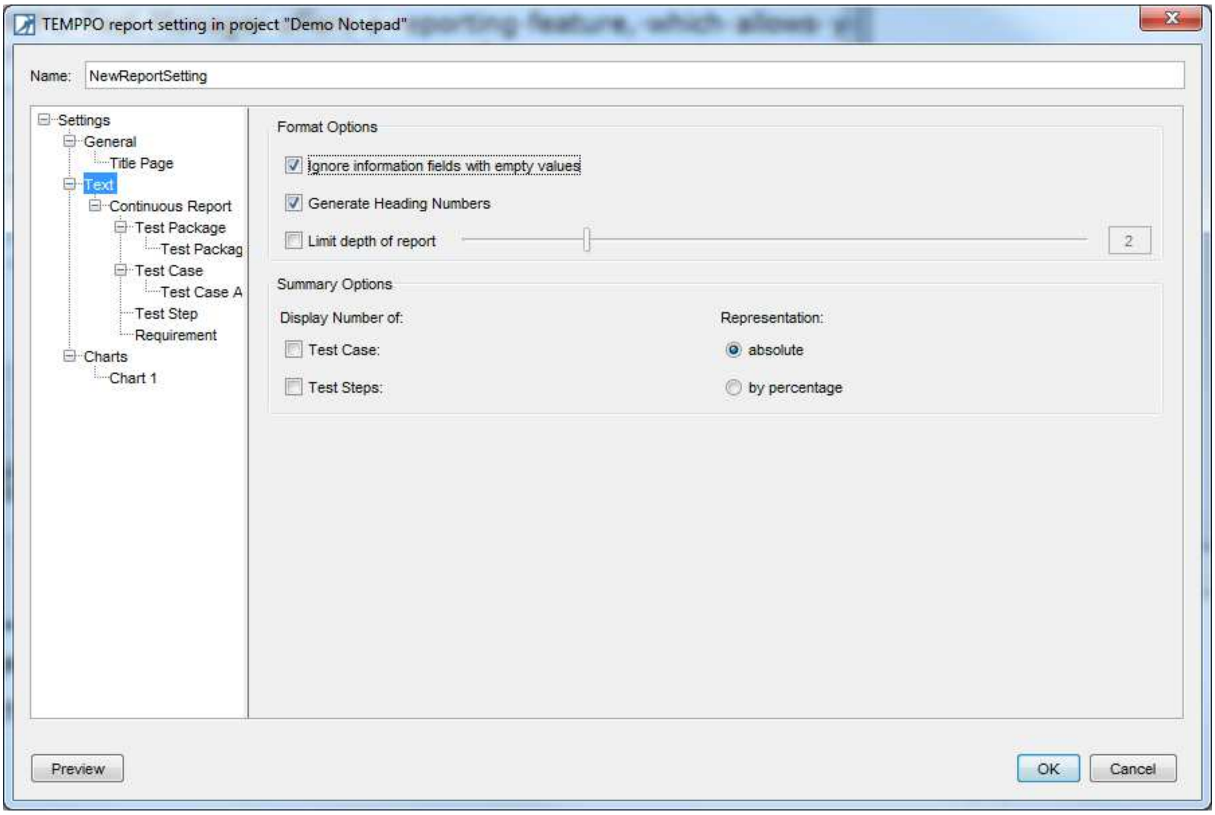
Furthermore you can set **Format options:**

- (Do not) ignore information fields with empty values
- (Do not) generate heading numbers

You can limit the size of your report by defining the depth absolutely. You can use the slider to limit the reported test structure or test suite tree depth to a fixed level.

You can add **Summary information**_about your test package by checking Number of Test cases and/or Test Steps. The amount of contained test cases/ test steps will be reported for each test package, either in absolute form or by percentage.

Finally you have the possibility to **Generate heading numbers** in your report.



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

Format Options

☑ Ignore information fields with empty values

☑ Generate Heading Numbers

□ Limit depth of report 2

Summary Options

Display Number of:	Representation:
□ Test Case:	● absolute
□ Test Steps:	○ by percentage

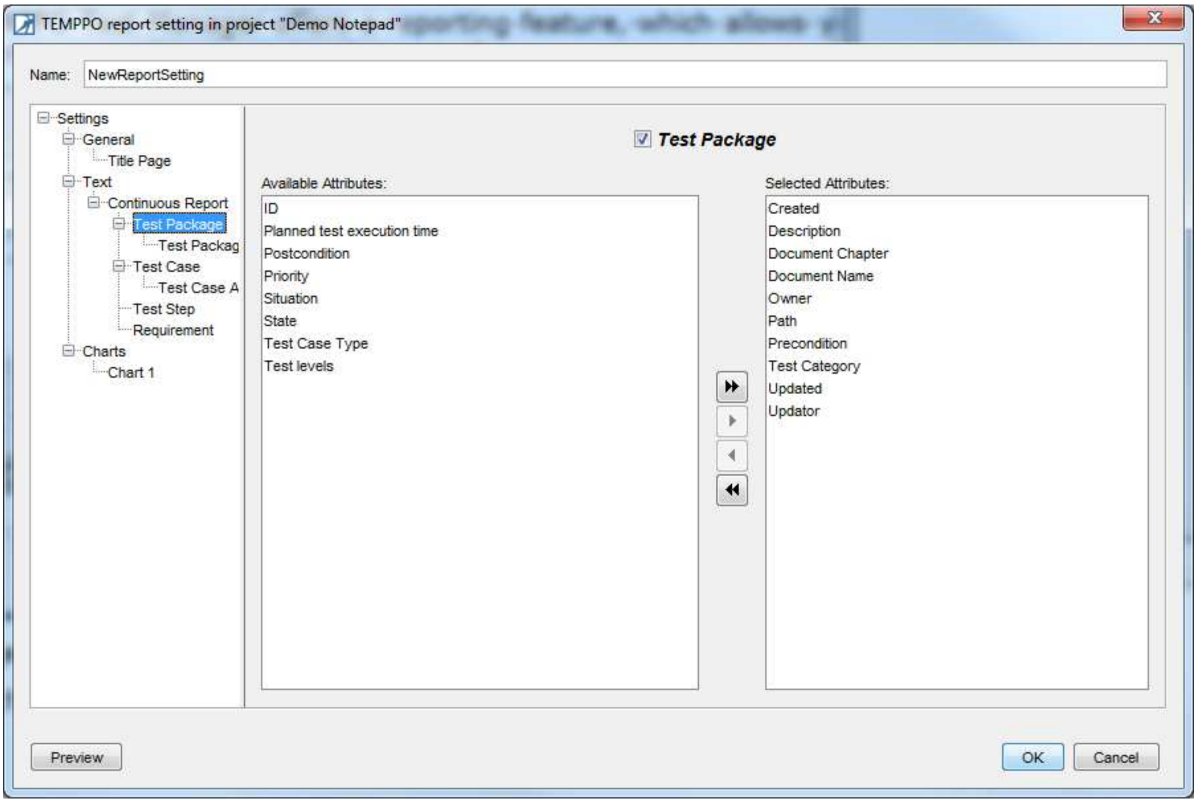
Figure 142 – Report: General Textual Settings

Continuous Report: Test Package, Test Case, Test Case Attributes, Test Result Attributes, Test Step, Requirements

For each reported object you can define, which data the report shall contain.

If you don’t want to show Test Packages, Test Cases, Test Case Attributes, Test Result Attributes, Test Steps or Requirements, click on the checkbox above the lists.

Multi select allows “moving” several values to the right part of the window.



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

- Settings
  - General
    - Title Page
  - Text
    - Continuous Report
      - Test Package
        - Test Packag
      - Test Case
        - Test Case A
      - Test Step
      - Requirement
  - Charts
    - Chart 1

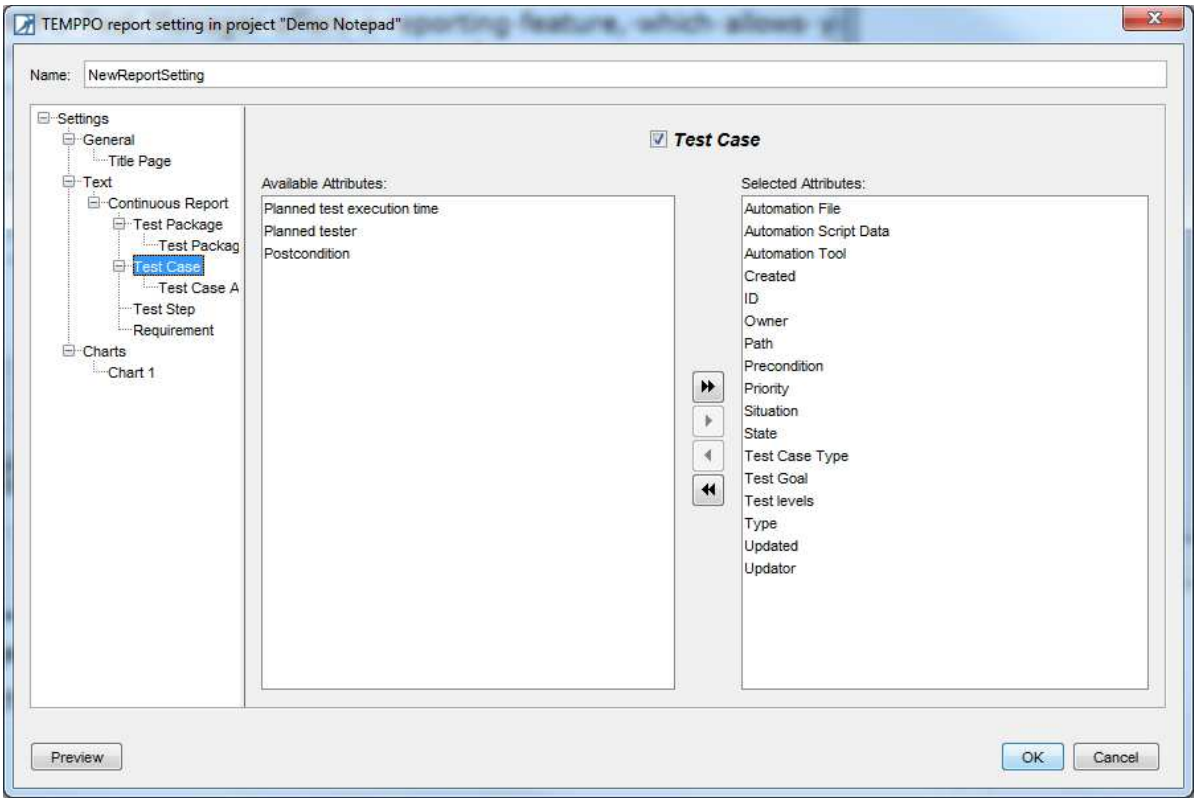
☑ **Test Package**

Available Attributes:

ID
Planned test execution time
Postcondition
Priority
Situation
State
Test Case Type
Test levels

Selected Attributes:

Created
Description
Document Chapter
Document Name
Owner
Path
Precondition
Test Category
Updated
Updator



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

- Settings
  - General
    - Title Page
  - Text
    - Continuous Report
      - Test Package
        - Test Packag
      - Test Case
        - Test Case A
      - Test Step
      - Requirement
  - Charts
    - Chart 1

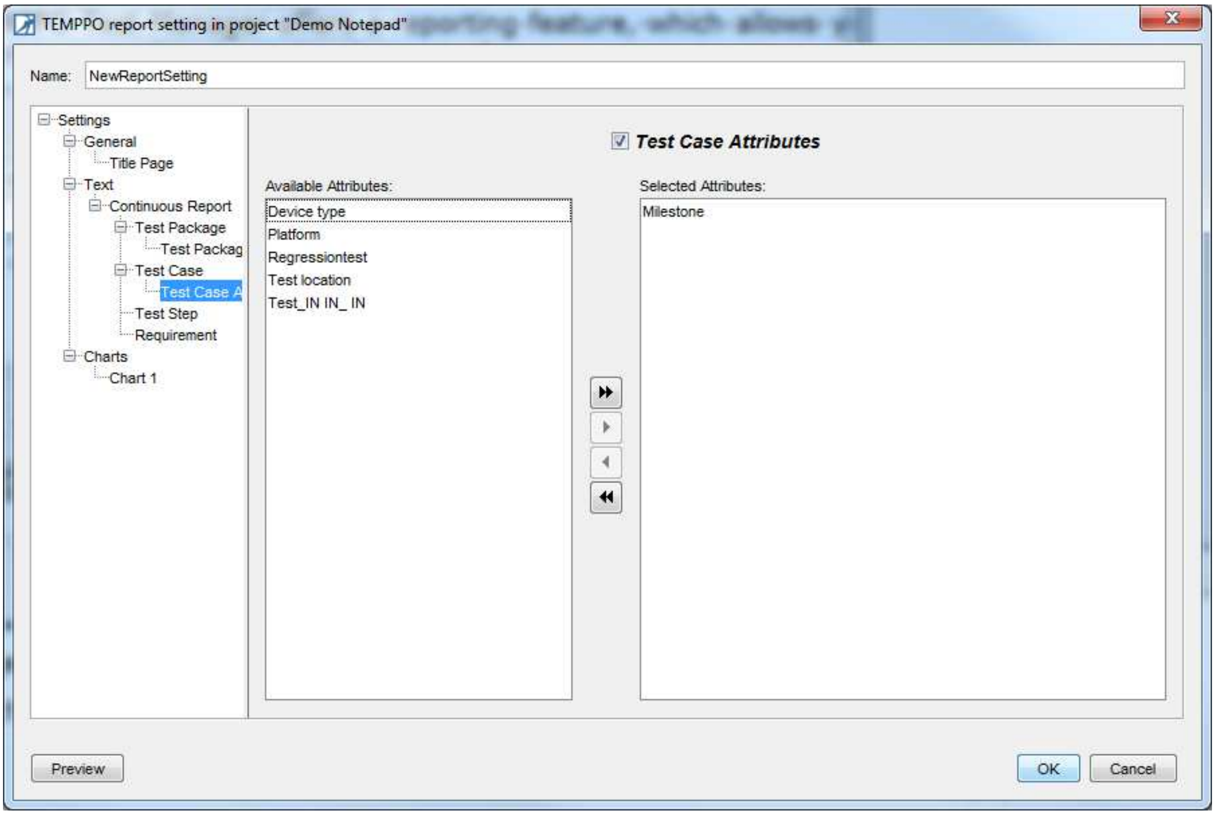
☑ **Test Case**

Available Attributes:

Planned test execution time
Planned tester
Postcondition

Selected Attributes:

Automation File
Automation Script Data
Automation Tool
Created
ID
Owner
Path
Precondition
Priority
Situation
State
Test Case Type
Test Goal
Test levels
Type
Updated
Updator



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

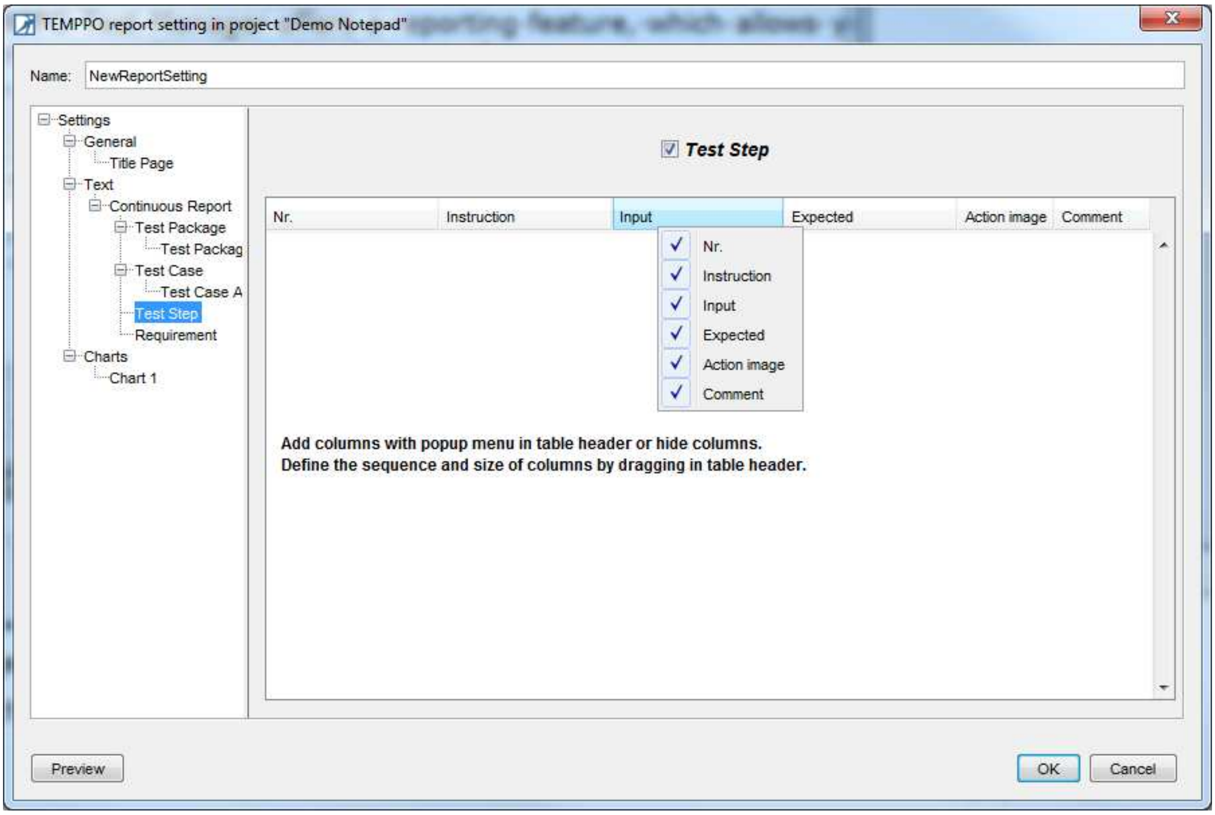
☑ **Test Case Attributes**

Available Attributes:

- Device type
- Platform
- Regressiontest
- Test location
- Test_IN IN_ IN

Selected Attributes:

- Milestone



TEMPPO report setting in project "Demo Notepad"

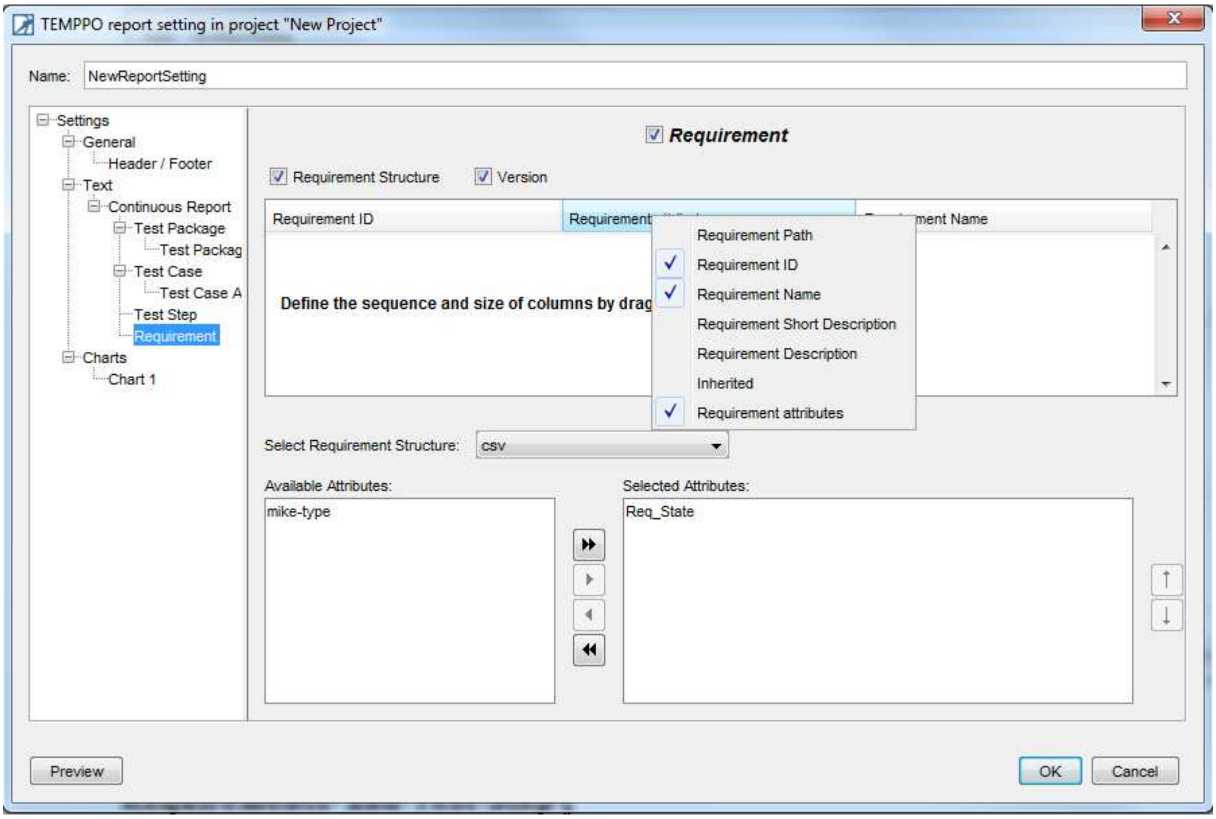
Name: NewReportSetting

☑ **Test Step**

Nr.	Instruction	Input	Expected	Action image	Comment

- ✓ Nr.
- ✓ Instruction
- ✓ Input
- ✓ Expected
- ✓ Action image
- ✓ Comment

Add columns with popup menu in table header or hide columns.
Define the sequence and size of columns by dragging in table header.



TEMPPO report setting in project "New Project"

Name: NewReportSetting

☑ Requirement

☑ Requirement Structure ☑ Version

Requirement ID	Requirement	ment Name

Define the sequence and size of columns by drag

Select Requirement Structure: csv

Available Attributes: mike-type

Selected Attributes: Req_State

Figure 143 – Report, Textual Settings

Tabular Report: Test Package, Test Case with its attributes, Requirements and Test Step

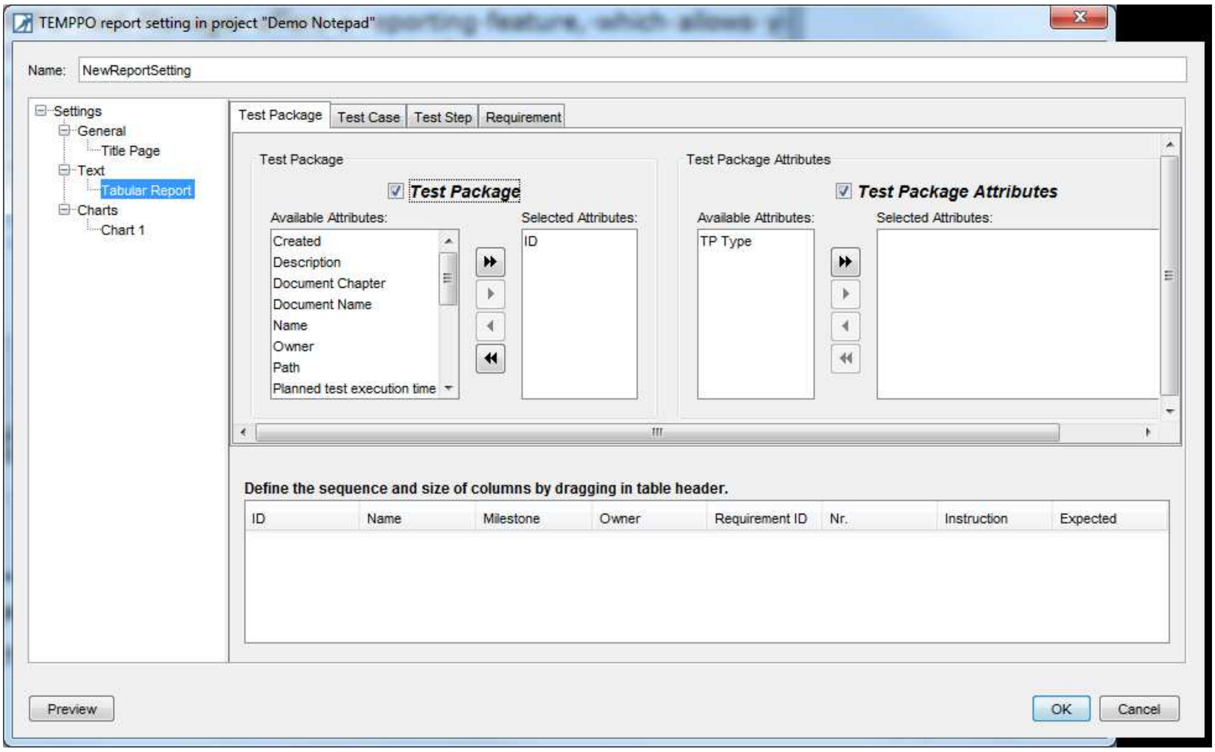
If you don’t want to show Test Packages, Test Cases, Test Steps click on the checkbox above the table.

Multi select allows “moving” several values to the right part of the window. For unselecting a column, double-click on it or open context menu and press remove.

Additionally you can define the sequence of columns by swapping the column headers and for defining the width of columns in the report you have to move the border of columns.

For each requirement structure the attributes can be reported selectively.

If you do not want to report requirement attributes in general right click on header and deselect “Requirement attributes”.



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

Test Package

☑ **Test Package**

Available Attributes: Created, Description, Document Chapter, Document Name, Name, Owner, Path, Planned test execution time

Selected Attributes: ID

Test Package Attributes

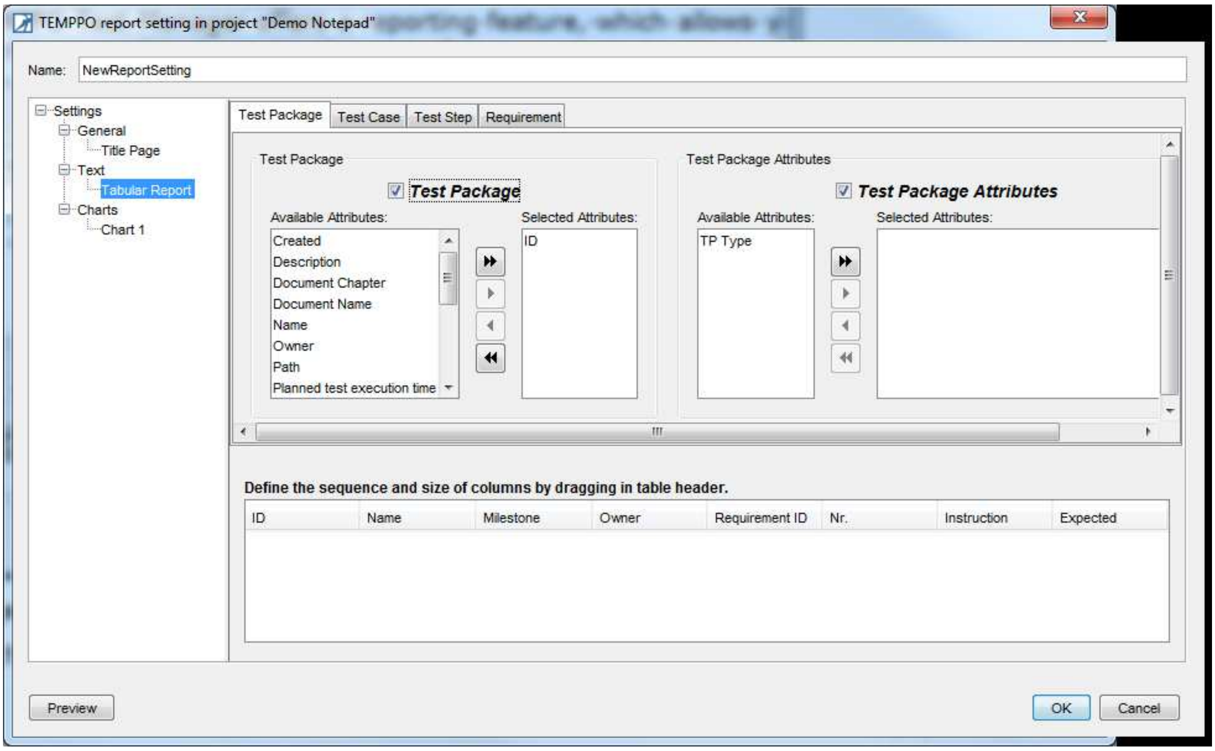
☑ **Test Package Attributes**

Available Attributes: TP Type

Selected Attributes:

Define the sequence and size of columns by dragging in table header.

ID	Name	Milestone	Owner	Requirement ID	Nr.	Instruction	Expected



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

Test Package

☑ **Test Package**

Available Attributes: Created, Description, Document Chapter, Document Name, Name, Owner, Path, Planned test execution time

Selected Attributes: ID

Test Package Attributes

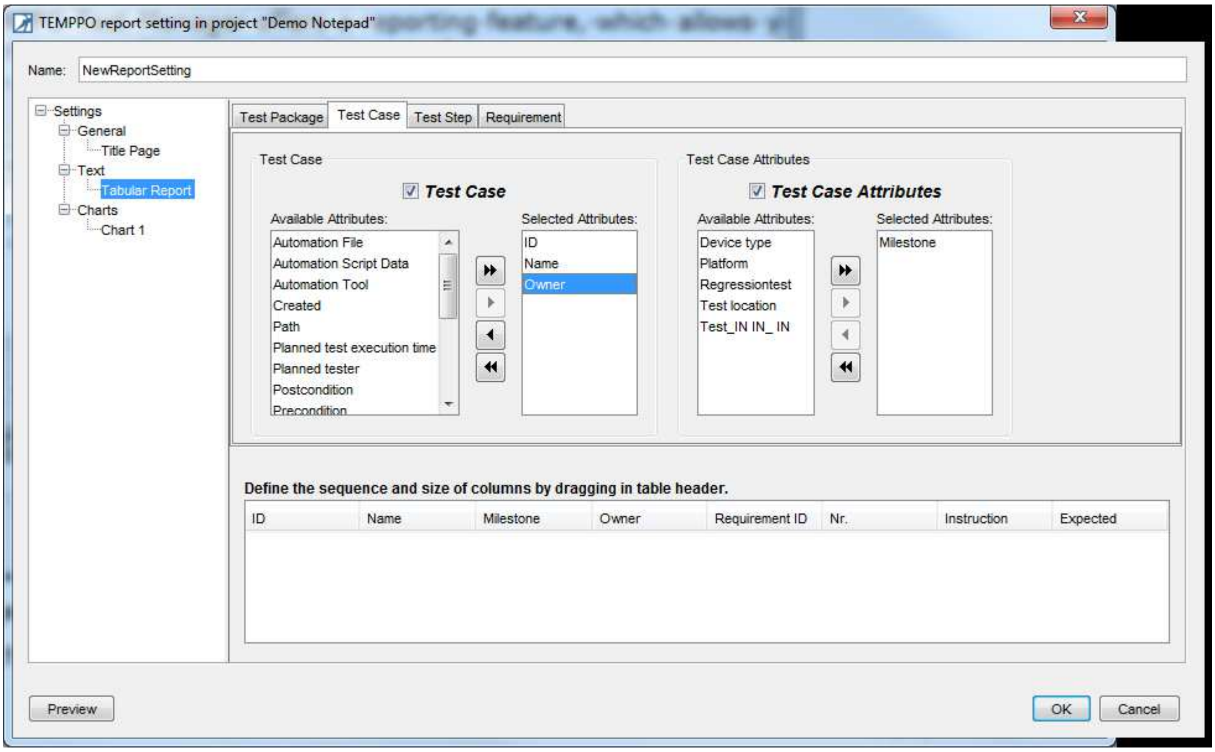
☑ **Test Package Attributes**

Available Attributes: TP Type

Selected Attributes:

Define the sequence and size of columns by dragging in table header.

ID | Name | Milestone | Owner | Requirement ID | Nr. | Instruction | Expected



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

- Settings
  - General
    - Title Page
  - Text
    - Tabular Report
  - Charts
    - Chart 1

Test Package | Test Case | Test Step | Requirement

Test Case

☑ **Test Case**

Available Attributes: Automation File, Automation Script Data, Automation Tool, Created, Path, Planned test execution time, Planned tester, Postcondition, Precondition

Selected Attributes: ID, Name, Owner

Test Case Attributes

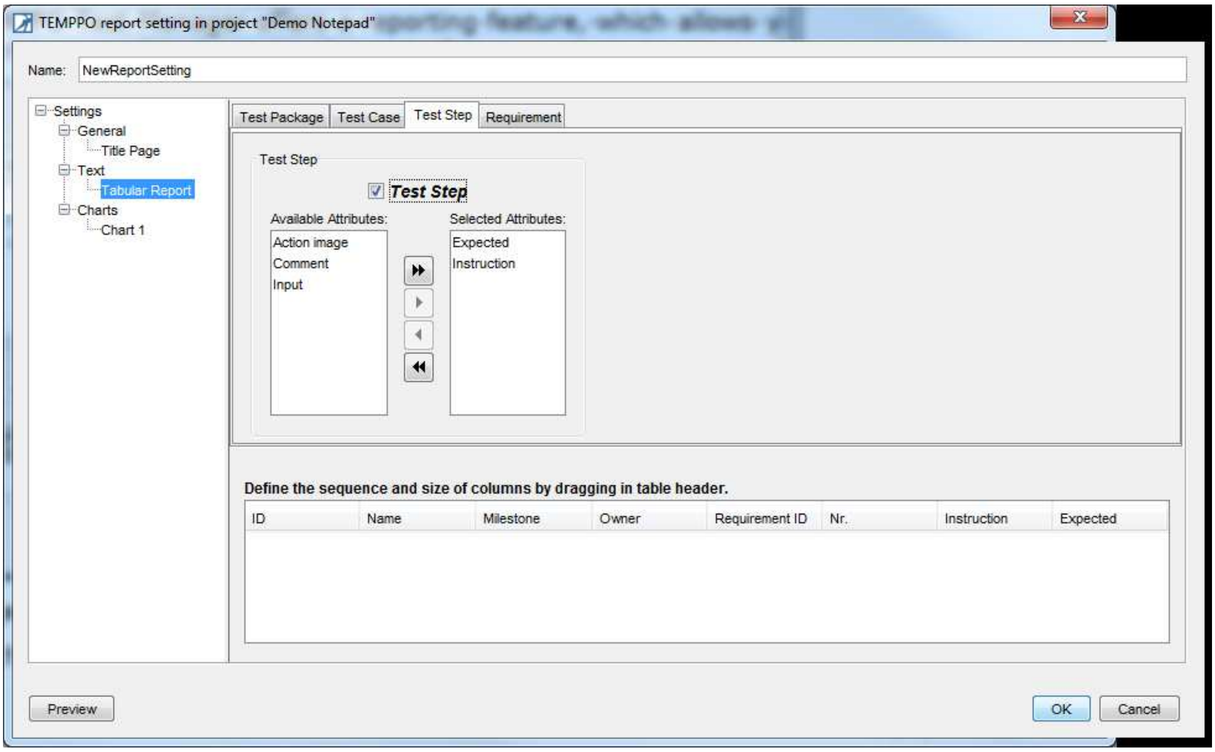
☑ **Test Case Attributes**

Available Attributes: Device type, Platform, Regressiontest, Test location, Test_IN IN_ IN

Selected Attributes: Milestone

Define the sequence and size of columns by dragging in table header.

ID	Name	Milestone	Owner	Requirement ID	Nr.	Instruction	Expected



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

- Settings
  - General
    - Title Page
  - Text
    - Tabular Report
  - Charts
    - Chart 1

Test Package | Test Case | Test Step | Requirement

Test Step

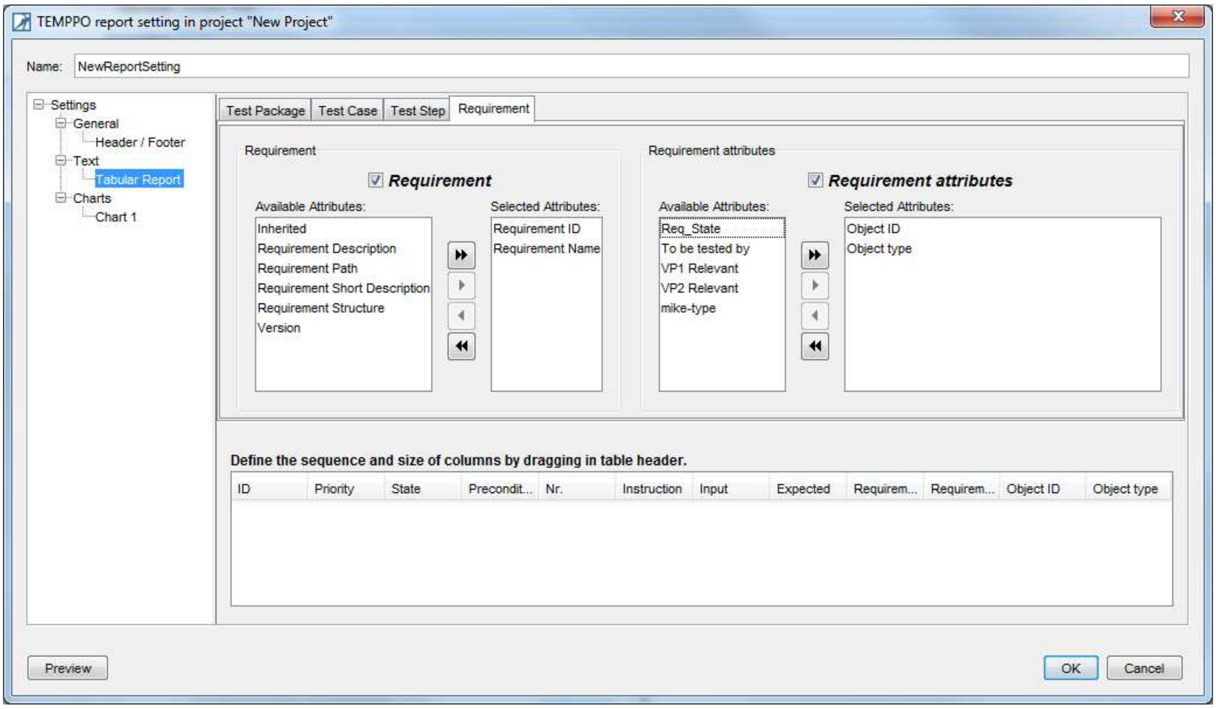
☑ **Test Step**

Available Attributes: Action image, Comment, Input

Selected Attributes: Expected, Instruction

Define the sequence and size of columns by dragging in table header.

ID	Name	Milestone	Owner	Requirement ID	Nr.	Instruction	Expected



TEMPPO report setting in project "New Project"

Name: NewReportSetting

- Settings
  - General
    - Header / Footer
  - Text
    - Tabular Report
  - Charts
    - Chart 1

Requirement

☑ ***Requirement***

Available Attributes: Inherited, Requirement Description, Requirement Path, Requirement Short Description, Requirement Structure, Version

Selected Attributes: Requirement ID, Requirement Name

Requirement attributes

☑ ***Requirement attributes***

Available Attributes: Req_State, To be tested by, VP1 Relevant, VP2 Relevant, mike-type

Selected Attributes: Object ID, Object type

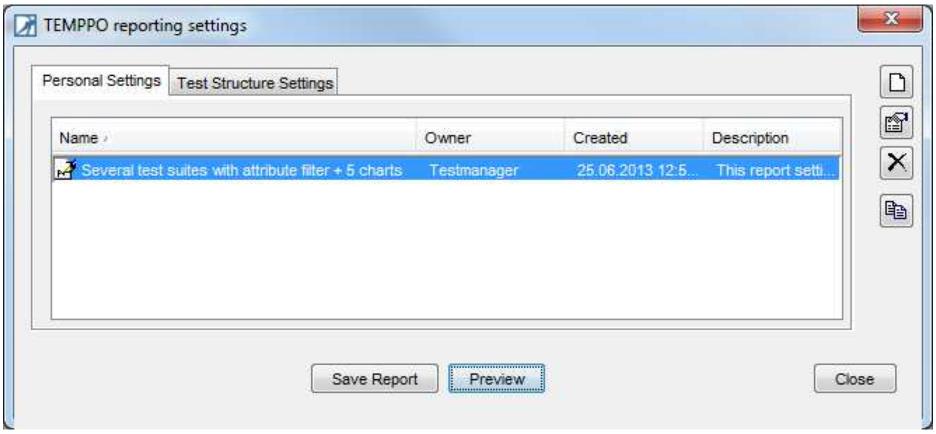
Define the sequence and size of columns by dragging in table header.

ID	Priority	State	Precondit...	Nr.	Instruction	Input	Expected	Requirem...	Requirem...	Object ID	Object type

Figure 144 – Tabular Report

3.1.5.6.2 Report of several test suites

When selecting the menu item **Evaluation > Report selected > Several Test Suites,** Figure 145 is displayed. Reports for several test suites are test structure wide. Therefore such a personal report setting can only be provided to other test structure users.



TEMPPO reporting settings

Name	Owner	Created	Description
Several test suites with attribute filter + 5 charts	Testmanager	25.06.2013 12:5...	This report setti...

Figure 145 - Report setting - test structure wide

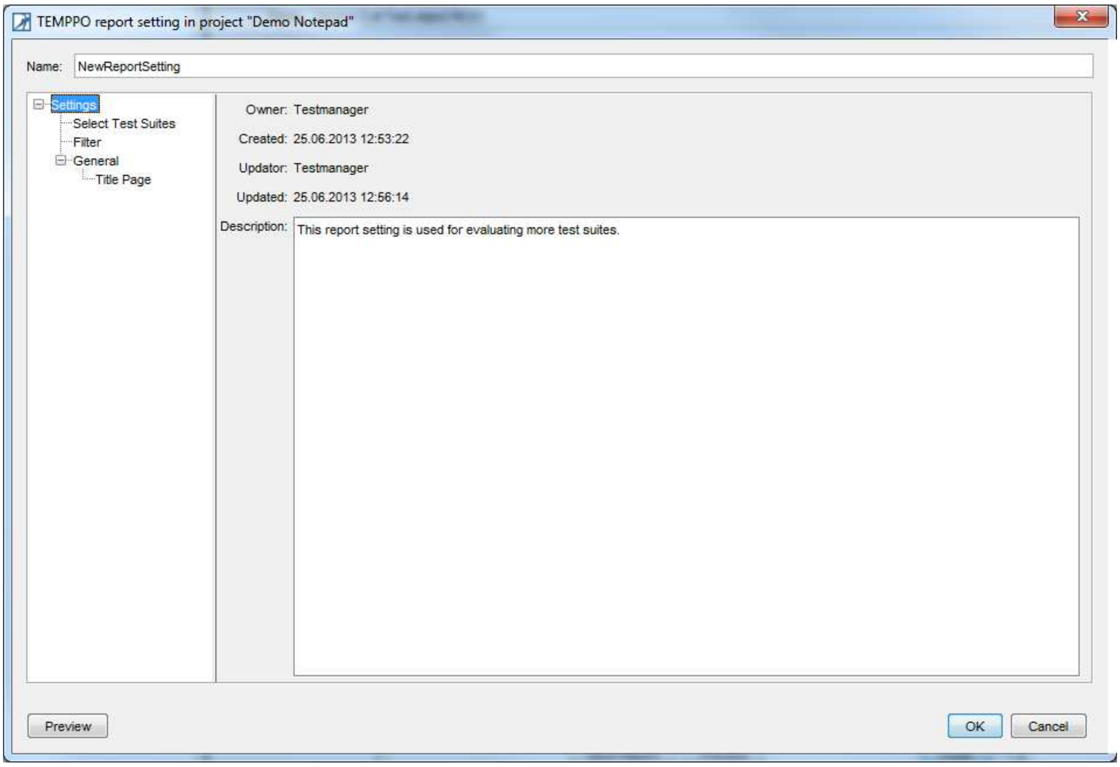
When creating or editing a report setting, Figure 146 is displayed, where you can

- Select the test suites of a test structure
- Set a filter for these test suites and
- Specify the report itself

Limitations: It is only possible

- to specify tabular reports as Excel or HTML.

- to select **Result – current** and **Result – current and history** for a test case



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

Owner: Testmanager

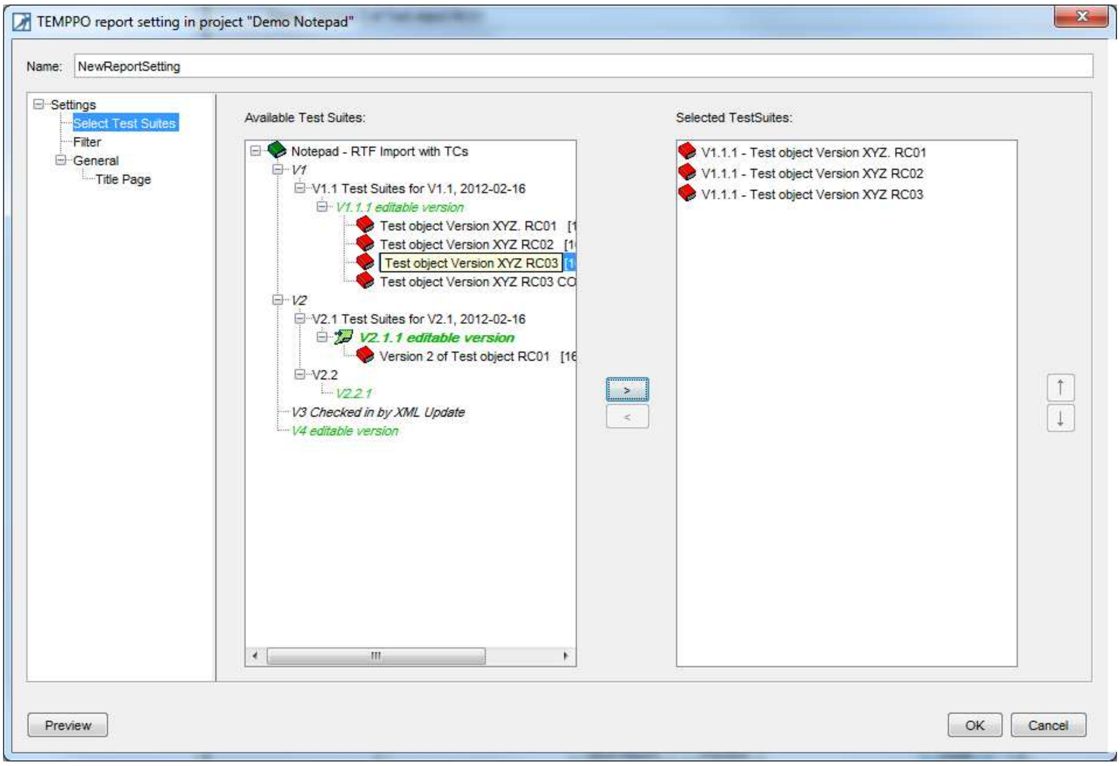
Created: 25.06.2013 12:53:22

Updator: Testmanager

Updated: 25.06.2013 12:56:14

Description: This report setting is used for evaluating more test suites.

Figure 146 - Report of several test suites: Settings



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

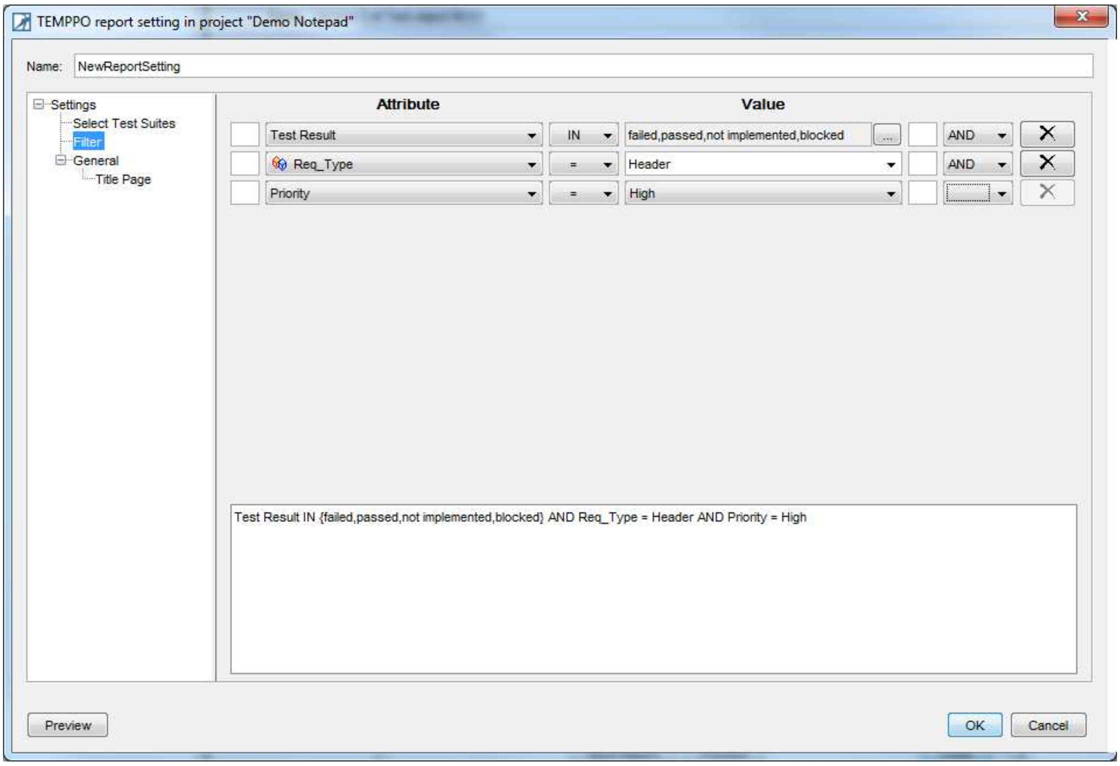
Available Test Suites:

Notepad - RTF Import with TCs
V1
V1.1 Test Suites for V1.1, 2012-02-16
V1.1.1 editable version
Test object Version XYZ. RC01 [1
Test object Version XYZ RC02 [1
Test object Version XYZ RC03 [1
Test object Version XYZ RC03 CO
V2
V2.1 Test Suites for V2.1, 2012-02-16
V2.1.1 editable version
Version 2 of Test object RC01 [16
V2.2
V2.2.1
V3 Checked in by XML Update
V4 editable version

Selected TestSuites:

V1.1.1 - Test object Version XYZ. RC01
V1.1.1 - Test object Version XYZ RC02
V1.1.1 - Test object Version XYZ RC03

Figure 147 - Report of several test suites: suite selection



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

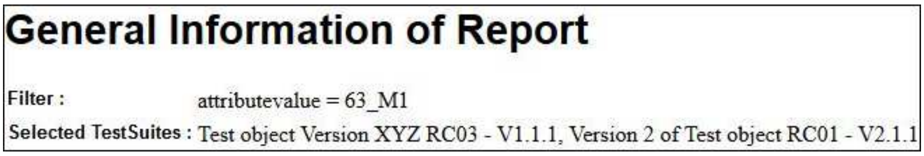
Attribute		Value	
Test Result	IN	failed,passed,not implemented,blocked	AND
Req_Type	=	Header	AND
Priority	=	High	

Test Result IN {failed,passed,not implemented,blocked} AND Req_Type = Header AND Priority = High

Figure 148 - Report of several test suites: Filter

Additionally to standard reports you select **General Information of Report** which displays the selected

- test suites
- selected filter

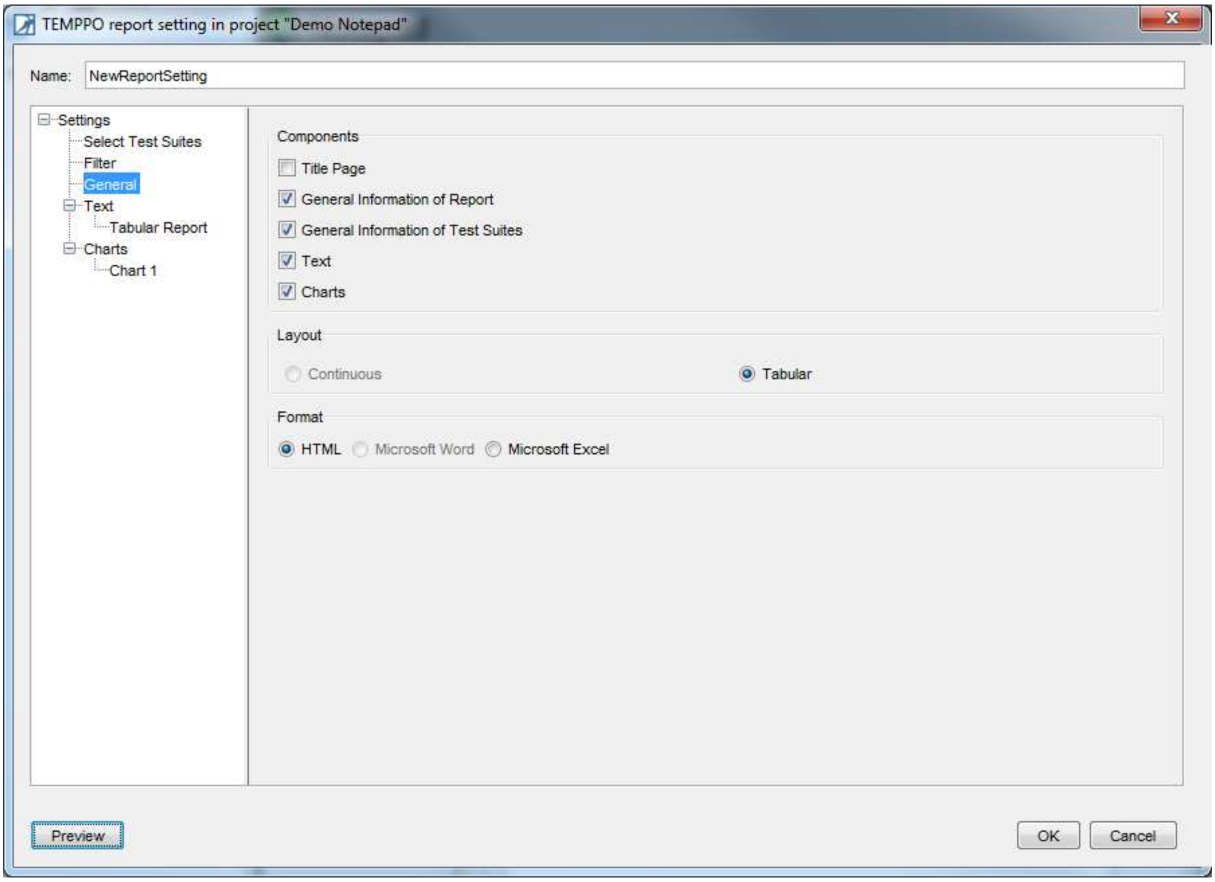


General Information of Report

Filter : attributevalue = 63_M1

Selected TestSuites : Test object Version XYZ RC03 - V1.1.1, Version 2 of Test object RC01 - V2.1.1

Figure 149 - Reported information



TEMPPO report setting in project "Demo Notepad"

Name: NewReportSetting

Components

□ Title Page
☑ General Information of Report
☑ General Information of Test Suites
☑ Text
☑ Charts

Layout

○ Continuous ● Tabular

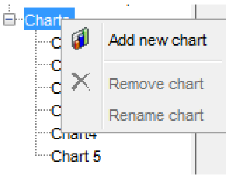
Format

● HTML ○ Microsoft Word ○ Microsoft Excel

Figure 150 - Report of several test suites: General

3.1.5.6.3 Chart Settings

In the charts settings area you can customize the charts added to your report, which represents the data of your test structure or your test suite in a graphical way.



It is possible to add more charts to your report by right mouse click on **chart** and activate **Add new chart**. By selecting a chart, right mouse click and activating **Remove chart**, a chart can be removed except the last one. By selecting a chart, right mouse click and activating **Rename chart** or **double click**, a char can be renamed.

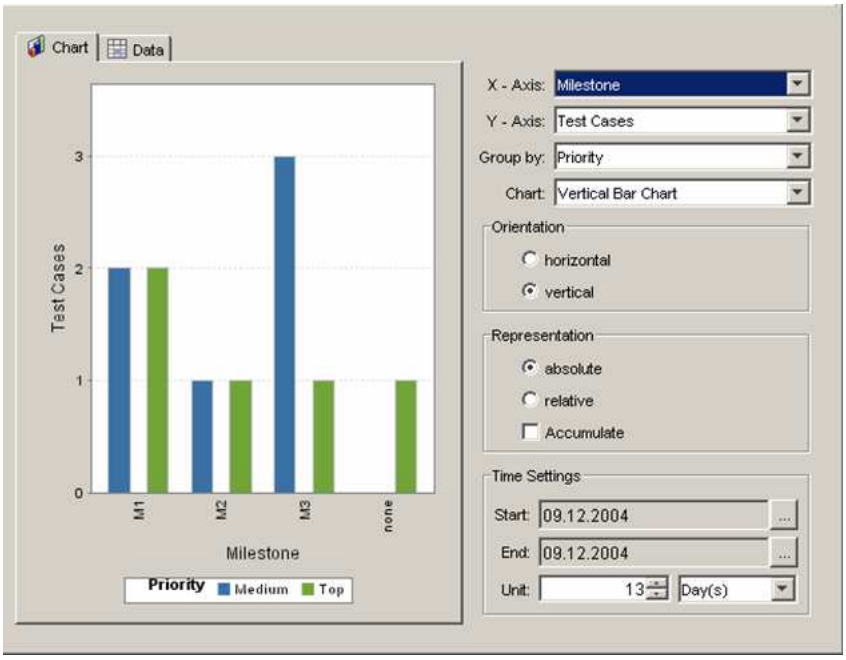
Data Selection:
Choose units for the y- and x-axis and optionally a value subdivision.

– ***Y-axis:*** sum of test packages, sum of test cases (default) or sum of test steps can be displayed.
– ***X-axis:*** test packages of first level, designer, date, priority, situation, state, type, test levels and result (only for test suite report) can be applied. If date is selected, the time period considered must be specified. Furthermore the time unit can be adjusted (day, week, month, year).
– ***Group by:*** is used for subdividing the values displayed on the y-axis. The selection possibilities are the same as for x-axis.
– **Chart:** The following charts are available: Horizontal Bar Chart, Horizontal Bar Chart (3D), Line Chart, Stacked Horizontal Bar Chart, Stacked Horizontal Bar Chart (3D), Stacked Vertical Bar Chart, Stacked Vertical Bar Chart (3D), Vertical Bar Chart, Vertical Bar Chart (3D)

Orientation:
You can choose the orientation of the labels on the x-axis.

Representation:
Data can be represented absolutely or relatively (use option buttons at bottom right). If *Accumulate* is activated, data on y-axis is displayed accumulated.

Time Settings:
If you select in x-axis and/or Group by a date / time attribute, you can configure the chart by setting a start and an end date. Additionally the units (days, weeks, months or years) can be set.



X - Axis: Milestone

Y - Axis: Test Cases

Group by: Priority

Chart: Vertical Bar Chart

Orientation

○ horizontal

● vertical

Representation

● absolute

○ relative

□ Accumulate

Time Settings

Start: 09.12.2004

End: 09.12.2004

Unit: 13 Day(s)

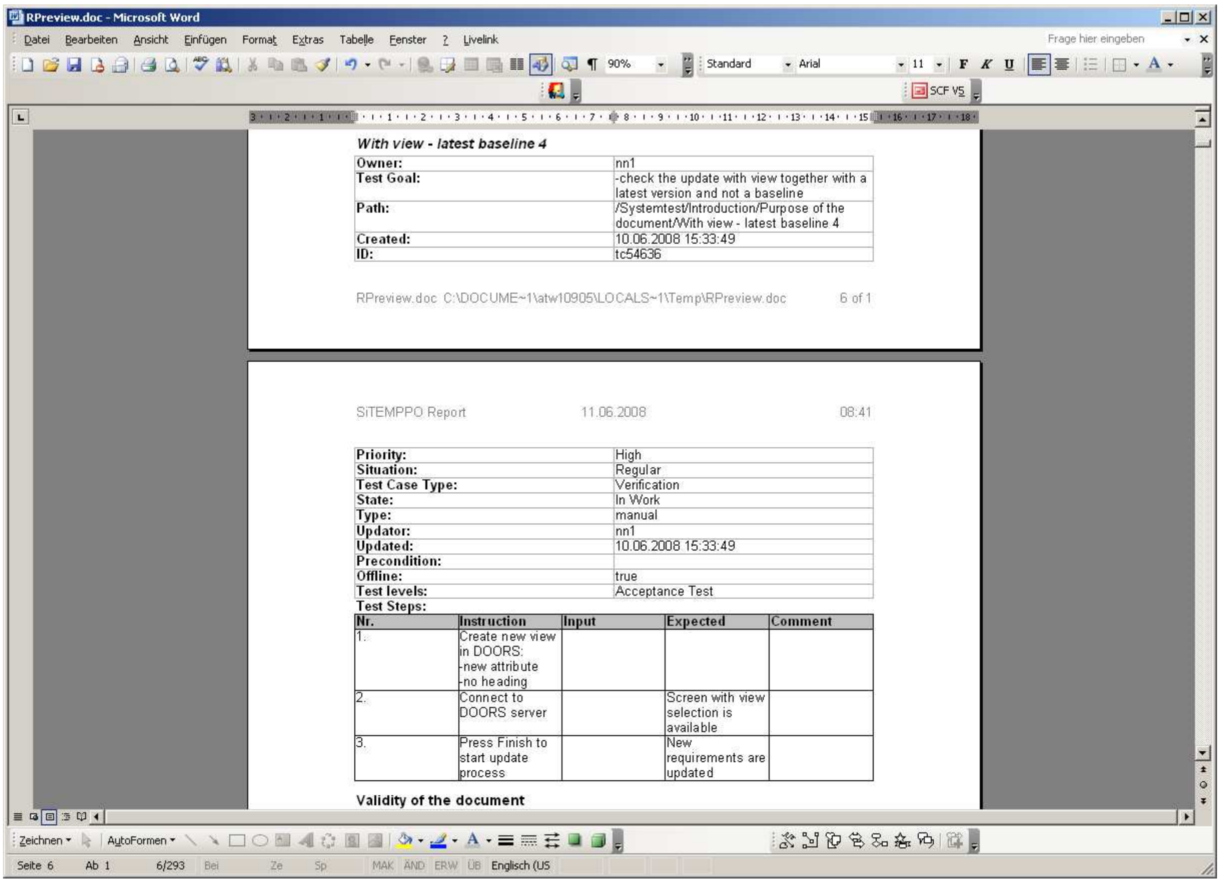
Figure 151 – Report: Chart Settings

3.1.5.6.4Previewing and Saving

You can either preview the report that will be generated or immediately save it to an Word or Excel-file (Button **Preview**).

ⓘ **Depending on the size of your test structure / test suite and network speed, the generation of the report (preview or file) may take a while, because a lot of data must be loaded from the database.**

Since TEMPPO 5.7 report preview is displayed in Microsoft Word or Microsoft Excel.


RPreview.doc - Microsoft Word
Datei Bearbeiten Ansicht Einfügen Format Extras Tabelle Fenster ? Livelink
Frage hier eingeben
90% Standard Arial 11
SCF VS
Zeichnen AutoFormen
Seite 6 Ab 1 6/293 Bei Ze Sp MAK ÄND ERW ÜB Englisch (US

With view - latest baseline 4

Owner:	nn1
Test Goal:	-check the update with view together with a latest version and not a baseline
Path:	/Systemtest/Introduction/Purpose of the document/With view - latest baseline 4
Created:	10.06.2008 15:33:49
ID:	tc54636

RPreview.doc C:\DOCUME~1\atw10905\LOCALS~1\Temp\RPreview.doc 6 of 1

SiTEMPPO Report 11.06.2008 08:41

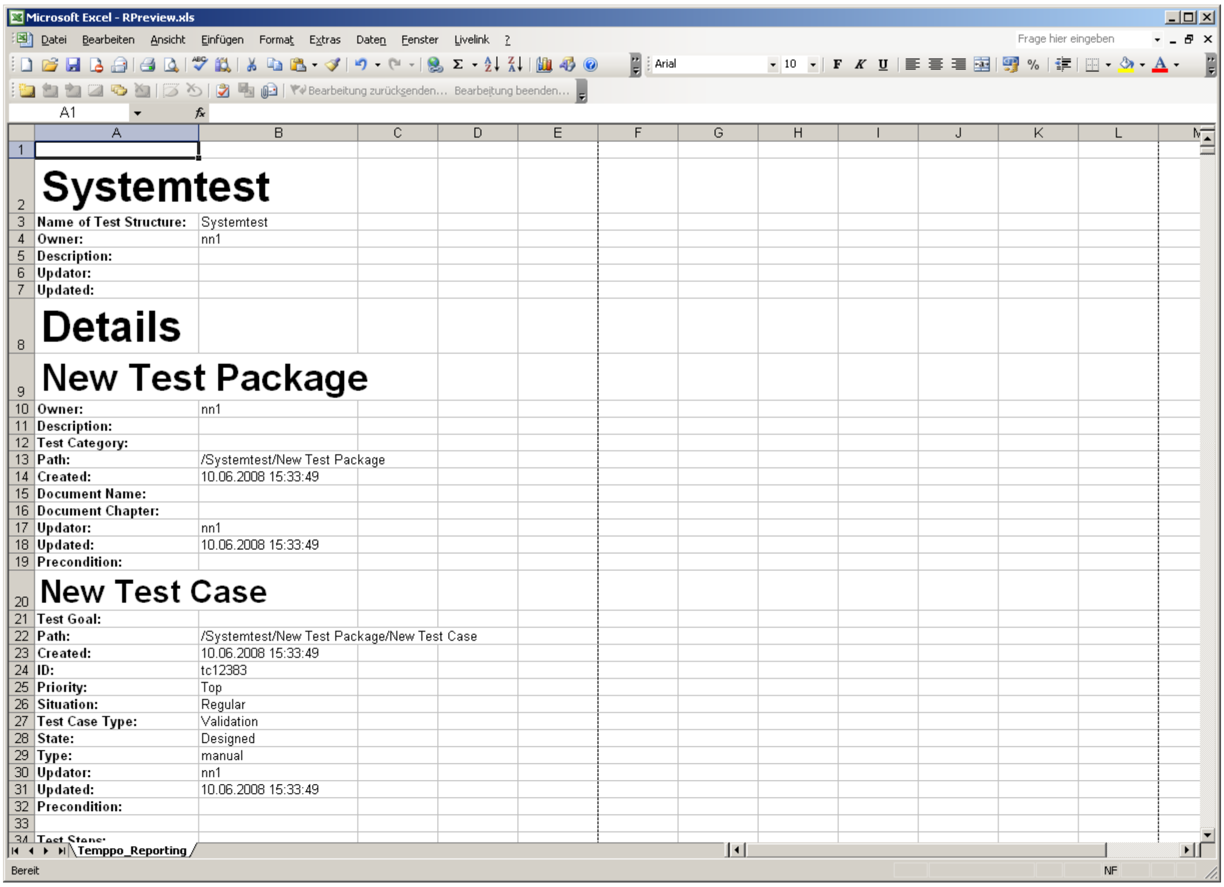
Priority:	High
Situation:	Regular
Test Case Type:	Verification
State:	In Work
Type:	manual
Updator:	nn1
Updated:	10.06.2008 15:33:49
Precondition:	
Offline:	true
Test levels:	Acceptance Test

Test Steps:

Nr.	Instruction	Input	Expected	Comment
1.	Create new view in DOORS: -new attribute -no heading			
2.	Connect to DOORS server		Screen with view selection is available	
3.	Press Finish to start update process		New requirements are updated	

Validity of the document

Figure 152 – Report: Word


Microsoft Excel - RPreview.xls
Datei Bearbeiten Ansicht Einfügen Format Extras Daten Fenster Livelink ?
Frage hier eingeben
Arial 10
Bearbeitung zurücksenden... Bearbeitung beenden...
A1
Temppo_Reporting
Bereit
NF

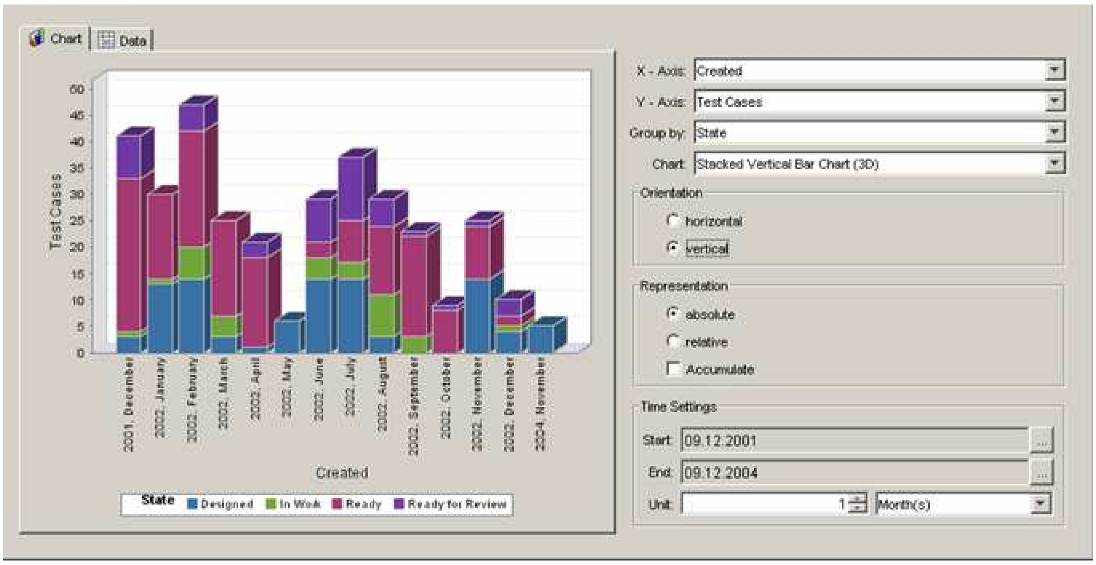
	A	B
1		
2	**Systemtest**	
3	**Name of Test Structure:**	Systemtest
4	**Owner:**	nn1
5	**Description:**	
6	**Updator:**	
7	**Updated:**	
8	**Details**	
9	**New Test Package**	
10	**Owner:**	nn1
11	**Description:**	
12	**Test Category:**	
13	**Path:**	/Systemtest/New Test Package
14	**Created:**	10.06.2008 15:33:49
15	**Document Name:**	
16	**Document Chapter:**	
17	**Updator:**	nn1
18	**Updated:**	10.06.2008 15:33:49
19	**Precondition:**	
20	**New Test Case**	
21	**Test Goal:**	
22	**Path:**	/Systemtest/New Test Package/New Test Case
23	**Created:**	10.06.2008 15:33:49
24	**ID:**	tc12383
25	**Priority:**	Top
26	**Situation:**	Regular
27	**Test Case Type:**	Validation
28	**State:**	Designed
29	**Type:**	manual
30	**Updator:**	nn1
31	**Updated:**	10.06.2008 15:33:49
32	**Precondition:**	
33		
34	**Test Steps:**	

Figure 153 – Report: Excel

3.1.5.6.5 Sample Charts

Test case creation progress (test structure):

– Graphical settings
  - **Y-axis**: Sum of test cases
  - **X-axis**: Created
  - **Group by**: State
  - **Date-Unit:** 1 Month
  - **Accumulate Data:** yes
  - **Display data:** absolute
  - **Chart Selection:** 3D Stacked Bar Chart



X - Axis: Created

Y - Axis: Test Cases

Group by: State

Chart: Stacked Vertical Bar Chart (3D)

Orientation
○ horizontal
● vertical

Representation
● absolute
○ relative
□ Accumulate

Time Settings
Start: 09.12.2001
End: 09.12.2004
Unit: 1 Month(s)

Figure 154 - Test case creation progress

Test case creation overview (test structure):

– Graphical settings
  - **Y-axis**: Sum of test cases
  - **X-axis**: Test packages of first level
  - **Value subdivision**: State
  - **Display data:** absolute
  - **Chart Selection:** 3D Stacked Bar Chart

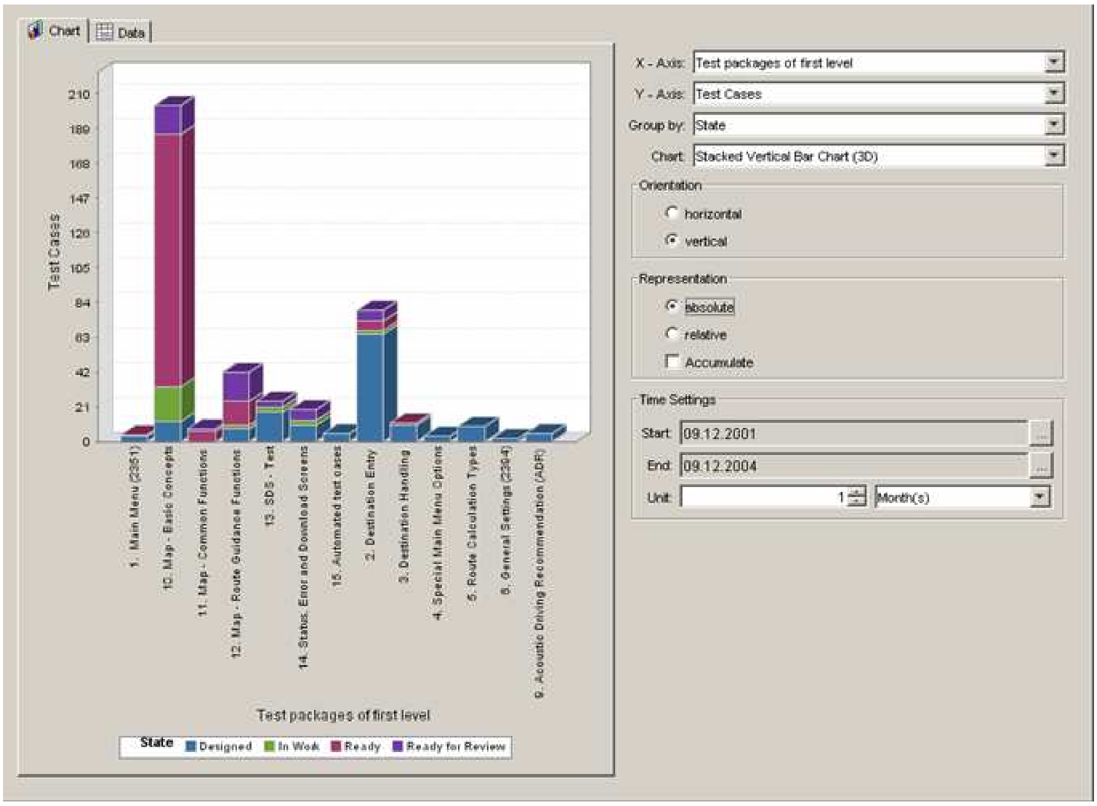


Chart
Data
210
189
168
147
126
105
84
63
42
21
0
Test Cases
1. Main Menu (2351)
10. Map - Basic Concepts
11. Map - Common Functions
12. Map - Route Guidance Functions
13. SDS - Test
14. Status, Error and Download Screens
15. Automated test cases
2. Destination Entry
3. Destination Handling
4. Special Main Menu Options
5. Route Calculation Types
8. General Settings (2304)
9. Acoustic Driving Recommendation (ADR)
Test packages of first level
State
Designed
In Work
Ready
Ready for Review

X - Axis: Test packages of first level

Y - Axis: Test Cases

Group by: State

Chart: Stacked Vertical Bar Chart (3D)

Orientation

○ horizontal

● vertical

Representation

● absolute

○ relative

☐ Accumulate

Time Settings

Start: 09.12.2001

End: 09.12.2004

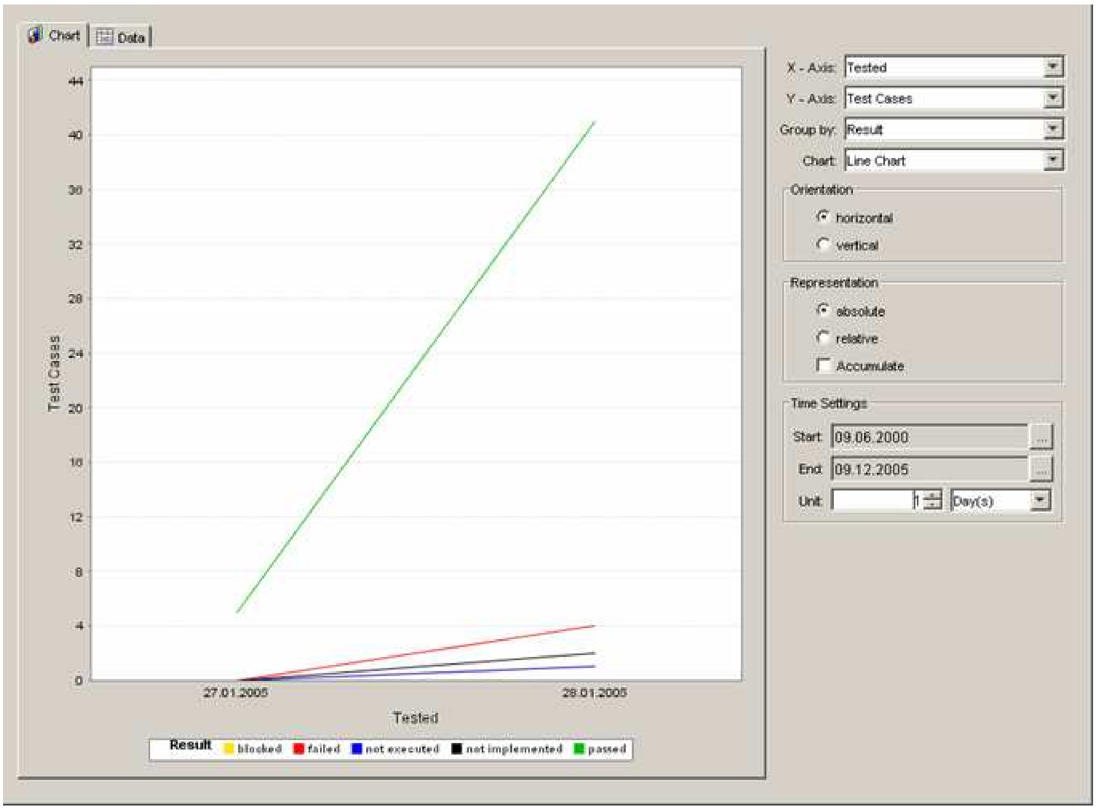
Unit: 1 Month(s)

Figure 155 - Test case creation overview

Test execution progress graph (test suite):

– Graphical settings

Y-axis:	Sum of test cases
X-axis:	Tested
Value subdivision:	Result
Date-Unit:	1 Day
Accumulate Data:	no
Display data:	absolute
Chart Selection:	Line Chart



X - Axis: Tested
Y - Axis: Test Cases
Group by: Result
Chart: Line Chart

Orientation
● horizontal
○ vertical

Representation
● absolute
○ relative
□ Accumulate

Time Settings
Start: 09.06.2000
End: 09.12.2005
Unit: 1 Day(s)

Figure 156 - Test execution progress graph (within one Test Suite)

3.1.5.6.6Report Templates

Some report templates are provided in your TEMPPO installation directory. These are mainly templates for

- Test plan and
- Test execution report

3.1.5.7 Import from another projects

You can import report settings from another project. You have to press the button Import report setting from another project then the following window opens:

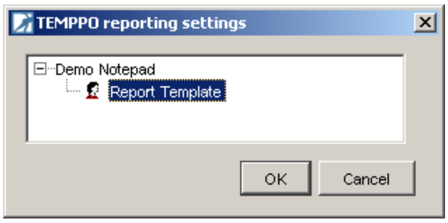


Figure 157 – Import from other project

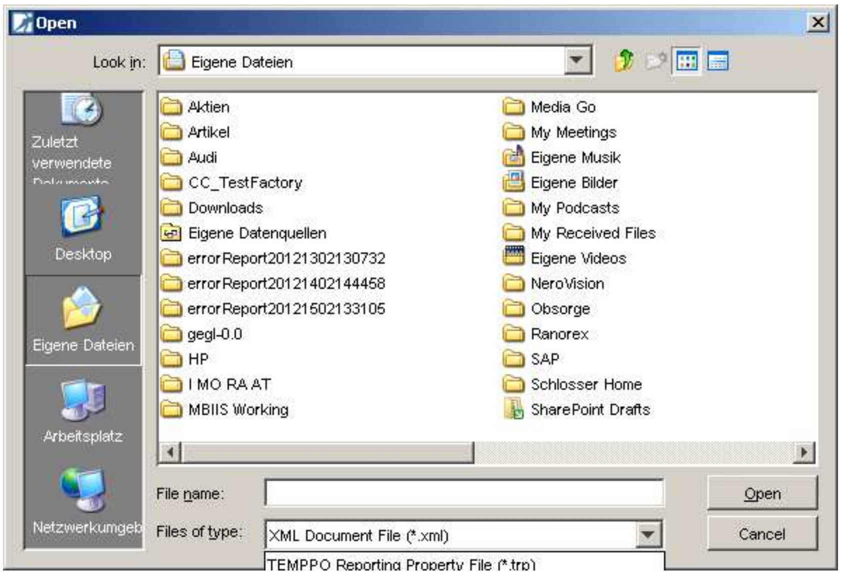
Select a report setting and then this setting is added to the actual setting.

3.1.5.8 Export to XML

For exchange you can export your report settings to an XML file. Press the button **Export report setting to XML file** and then a file dialog opens. After selecting the path and file name, press **Save**. Then the file is exported to XML.

3.1.5.9 Import from XML / TRP file

For importing report settings, press the button **Import report setting from XML or TRP file** and then the following file dialog opens:



Look in: Eigene Dateien

Files of type: XML Document File (*.xml)

Figure 158 – Import report settings

In the file dialog you can set the type of file to .trp or .xml. After selecting a file and pressing open, report setting is imported to TEMPPO.

3.2 Multi user ability

Multi user ability establishes an environment for working with several users in a parallel way on the same test structure or suite. It prevents unintentional overriding of changes, manages the synchronization between viewed and stored data and avoids inconsistency, if multiple users work on the same database. Moreover it provides features to facilitate the collaboration in a team working on the same TEMPPO Project.

Chapter 3.2.1 describes the behavior when locking and editing an item. On the other hand chapter 3.2.2 illustrates the 2 possibilities of refreshing (manual, automatic).

ⓘ **Multi user ability is not supported for MS Access databases.**

3.2.1 Lock modes

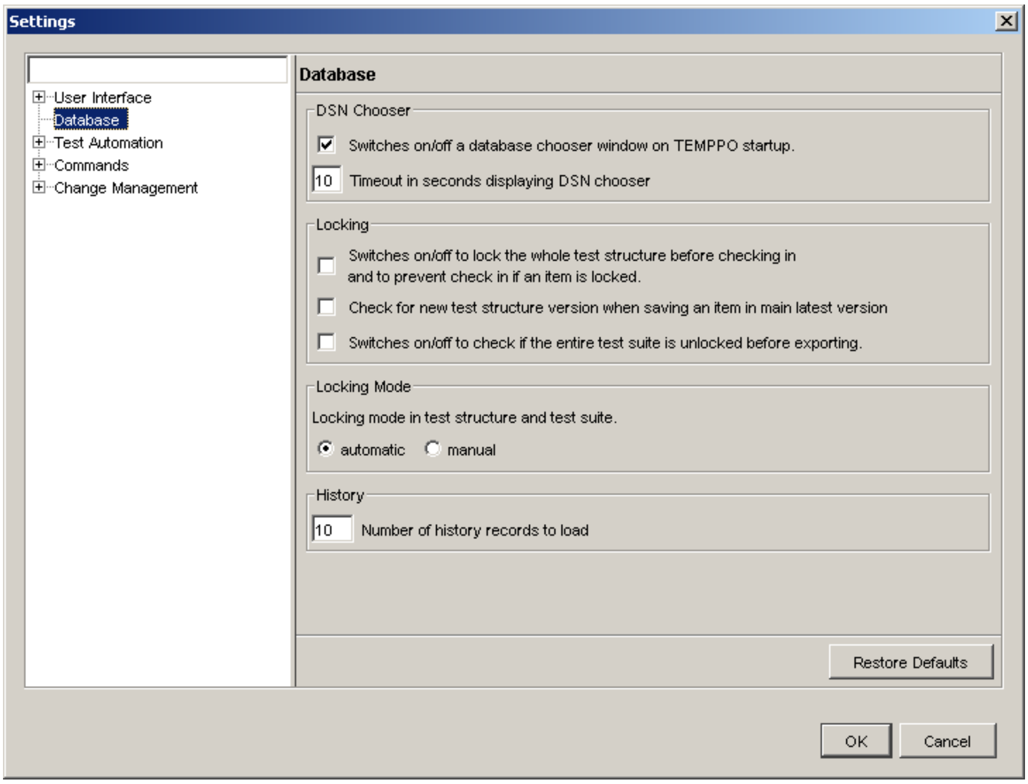
TEMPPO provides 2 possibilities of locking items (test packages, test cases):

- **Manual lock**
- **Automatic lock**

If you are working with manual lock mode, you always have to press the button **Lock** before having the exclusive right to work on that item.

On the other hand you simply have to select an item and it is automatically locked, if another user does not lock it.

This feature can be configured in TEMPPO settings:



Database

DSN Chooser

☑ Switches on/off a database chooser window on TEMPPO startup.

10 Timeout in seconds displaying DSN chooser

Locking

□ Switches on/off to lock the whole test structure before checking in and to prevent check in if an item is locked.

□ Check for new test structure version when saving an item in main latest version

□ Switches on/off to check if the entire test suite is unlocked before exporting.

Locking Mode

Locking mode in test structure and test suite.

● automatic ○ manual

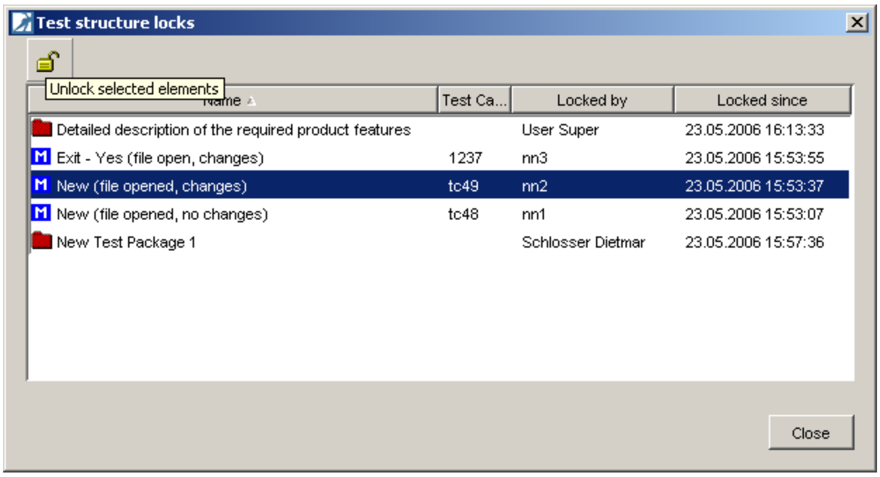
History

10 Number of history records to load

Figure 159 - Settings->Database->Locking

The following chapters are describing the locking in manual mode!

3.2.1.1 Find locks – unlock



Name	Test Ca...	Locked by	Locked since
Detailed description of the required product features		User Super	23.05.2006 16:13:33
Exit - Yes (file open, changes)	1237	nn3	23.05.2006 15:53:55
New (file opened, changes)	tc49	nn2	23.05.2006 15:53:37
New (file opened, no changes)	tc48	nn1	23.05.2006 15:53:07
New Test Package 1		Schlosser Dietmar	23.05.2006 15:57:36

Figure 160 - Test structure - Find locks

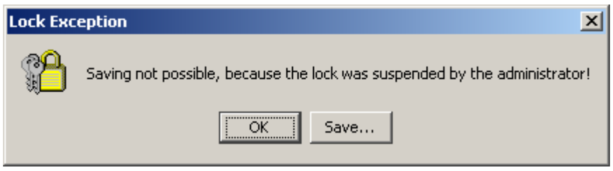
In principle TEMPPO differs between 2 locks: User locks (e.g. select a TP, TC) and locks caused from exports (see 3.1.4.7.1). The following lock types are possible:

- Test case
- Test package
- Test result (test case in test suite)
- Test suite (root)
- Test structure (root)
- Version

Locks can be found either in the test structure or test suite

- Test structure:
  When activating the menu item **Find locks** (menu test structure), Figure 160 is displayed. By selecting one or more locks and pressing the button **Unlock selected elements** locks can be removed.
- Test suite:
  When activating the menu item **Find locks** (menu test suite), Figure 160 is displayed. By selecting one or more locks and pressing the button **Unlock selected elements** locks can be removed.

If the corresponding user that still believes in his locked items presses the button **Save**, he will get the following message, that the lock have be removed in the meantime.



Saving not possible, because the lock was suspended by the administrator!

Figure 161 - Lock deletion message

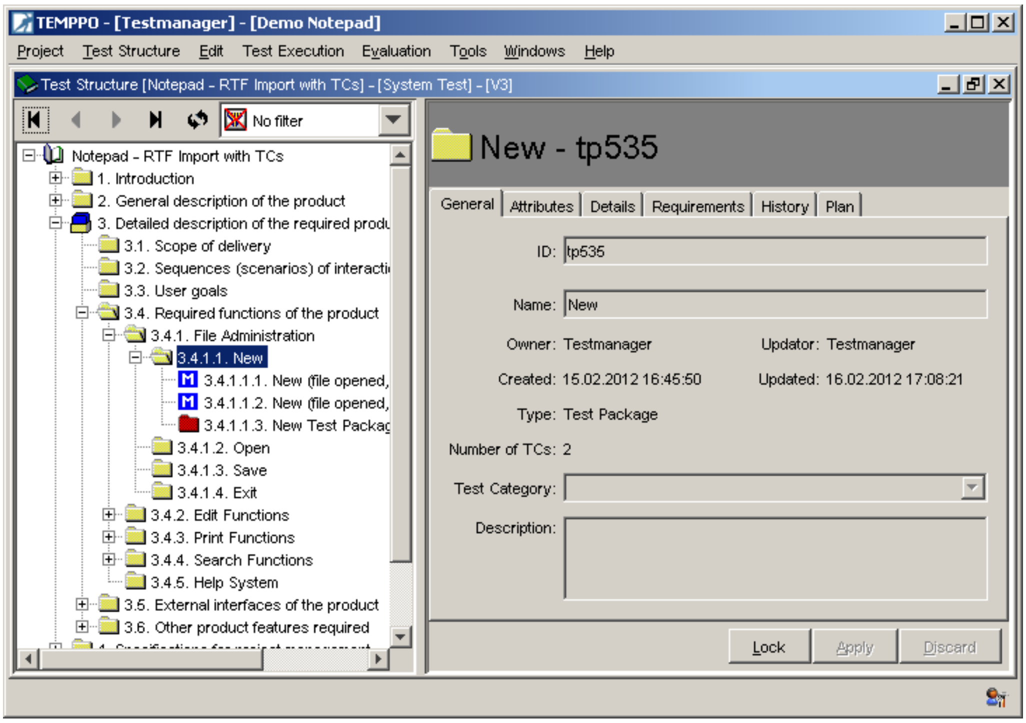
When closing the window by pressing OK, he can only discard his changes!

3.2.1.2 Showing a lock state

The locking state of the item is shown by:

Special padlock icon (green: locked by the current user, yellow: locked by another user or due to an exported test suite).

ⓘ **Tool Tip "Locked by <user>**



TEMPPO - [Testmanager] - [Demo Notepad]

Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]

- Notepad - RTF Import with TCs
  - 1. Introduction
  - 2. General description of the product
  - 3. Detailed description of the required produ
    - 3.1. Scope of delivery
    - 3.2. Sequences (scenarios) of interacti
    - 3.3. User goals
    - 3.4. Required functions of the product
      - 3.4.1. File Administration
        - 3.4.1.1. New
          - 3.4.1.1.1. New (file opened,
          - 3.4.1.1.2. New (file opened,
          - 3.4.1.1.3. New Test Packag
        - 3.4.1.2. Open
        - 3.4.1.3. Save
        - 3.4.1.4. Exit
      - 3.4.2. Edit Functions
      - 3.4.3. Print Functions
      - 3.4.4. Search Functions
      - 3.4.5. Help System
    - 3.5. External interfaces of the product
    - 3.6. Other product features required

New - tp535

ID: tp535

Name: New

Owner: Testmanager Updator: Testmanager

Created: 15.02.2012 16:45:50 Updated: 16.02.2012 17:08:21

Type: Test Package

Number of TCs: 2

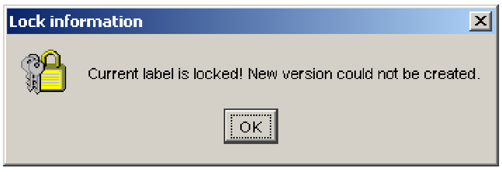
Test Category:

Description:

Figure 162 - Lock button

3.2.1.3 Creating a version

There a different ways for creating a new version, e.g. check in of a version or creating a new branch (for test suite). During the creation process the current version is locked so that no other user can create a version based on the same version. Is the version already locked, the following dialog is shown.



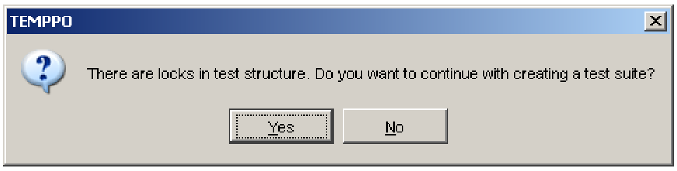
Lock information

Current label is locked! New version could not be created.

Figure 163 – Version locked

3.2.1.4 Creating a test suite

You can only create a new test suite, if the test structure and all of its items are not locked. Otherwise you cannot create a test suite and you will get the following dialog (see Figure 164), after pressing the **Finish** button on the test suite wizard:



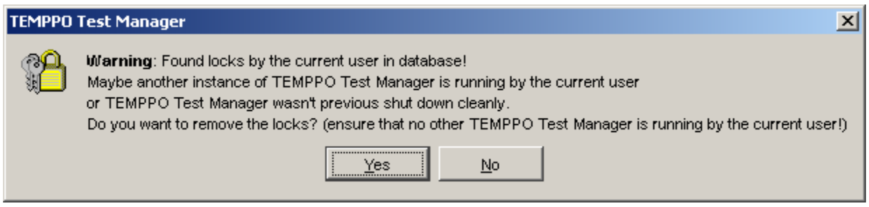
TEMPPO

There are locks in test structure. Do you want to continue with creating a test suite?

Figure 164 - Test structure locked

3.2.1.5 Finding invalid locks

On starting and quitting TEMPPO, the whole database is checked for any locks which were set by you. Normally there should be no one left. But if there is one (see Figure 165), this is an invalid lock, which can have two reasons: Either you are currently running a second TEMPPO Test Manager, or your last TEMPPO session terminated abnormally (crash etc.). If so, you should confirm the dialog by pressing “Yes” to remove all invalid locks.



TEMPPO Test Manager

Warning: Found locks by the current user in database!
Maybe another instance of TEMPPO Test Manager is running by the current user
or TEMPPO Test Manager wasn't previous shut down cleanly.
Do you want to remove the locks? (ensure that no other TEMPPO Test Manager is running by the current user!)

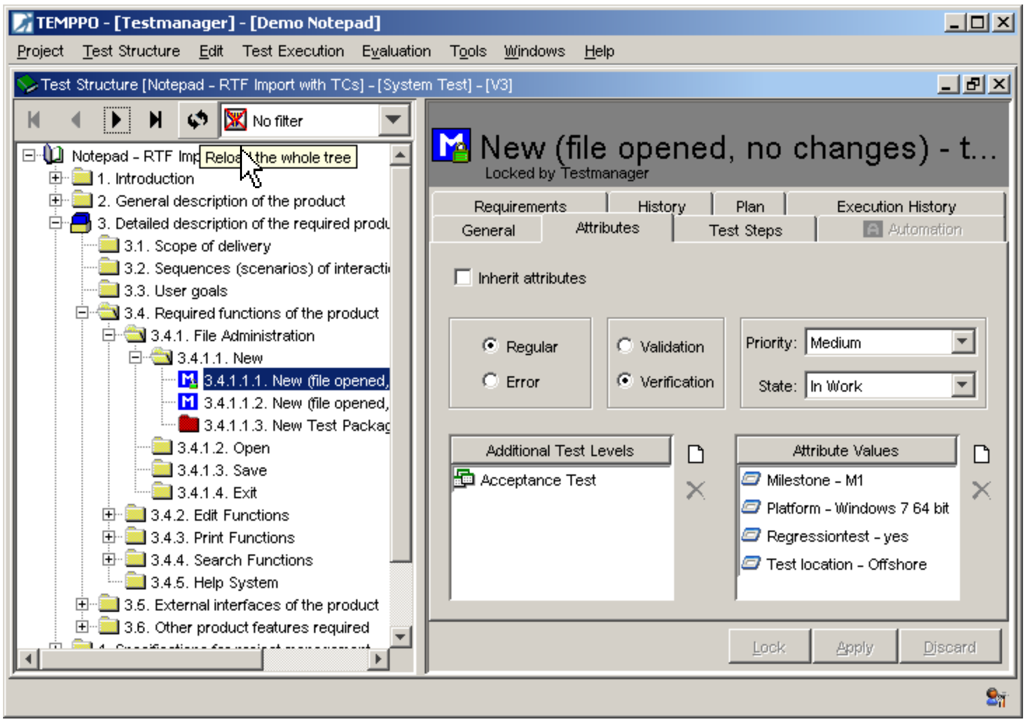
Figure 165 – Locks found in the database on starting/quitting TEMPPO

3.2.2 Refreshing

3.2.2.1 Refreshing manually

3.2.2.1.1 Reloading the whole test structure / test suite

You can reload the whole test structure / test suite by pressing the reload button ⟳ on top of the tree (see Figure 166). This has the same effect as manually reopening the test structure / test suite.



TEMPPO - [Testmanager] - [Demo Notepad]

Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]

No filter

Reload the whole tree

- Notepad - RTF Imp
  - 1. Introduction
  - 2. General description of the product
  - 3. Detailed description of the required produ
    - 3.1. Scope of delivery
    - 3.2. Sequences (scenarios) of interacti
    - 3.3. User goals
    - 3.4. Required functions of the product
      - 3.4.1. File Administration
        - 3.4.1.1. New
          - 3.4.1.1.1. New (file opened,
          - 3.4.1.1.2. New (file opened,
          - 3.4.1.1.3. New Test Packag
        - 3.4.1.2. Open
        - 3.4.1.3. Save
        - 3.4.1.4. Exit
      - 3.4.2. Edit Functions
      - 3.4.3. Print Functions
      - 3.4.4. Search Functions
      - 3.4.5. Help System
    - 3.5. External interfaces of the product
    - 3.6. Other product features required

New (file opened, no changes) - t...
Locked by Testmanager

☐ Inherit attributes

● Regular
○ Error

○ Validation
● Verification

Priority: Medium
State: In Work

Additional Test Levels
- Acceptance Test

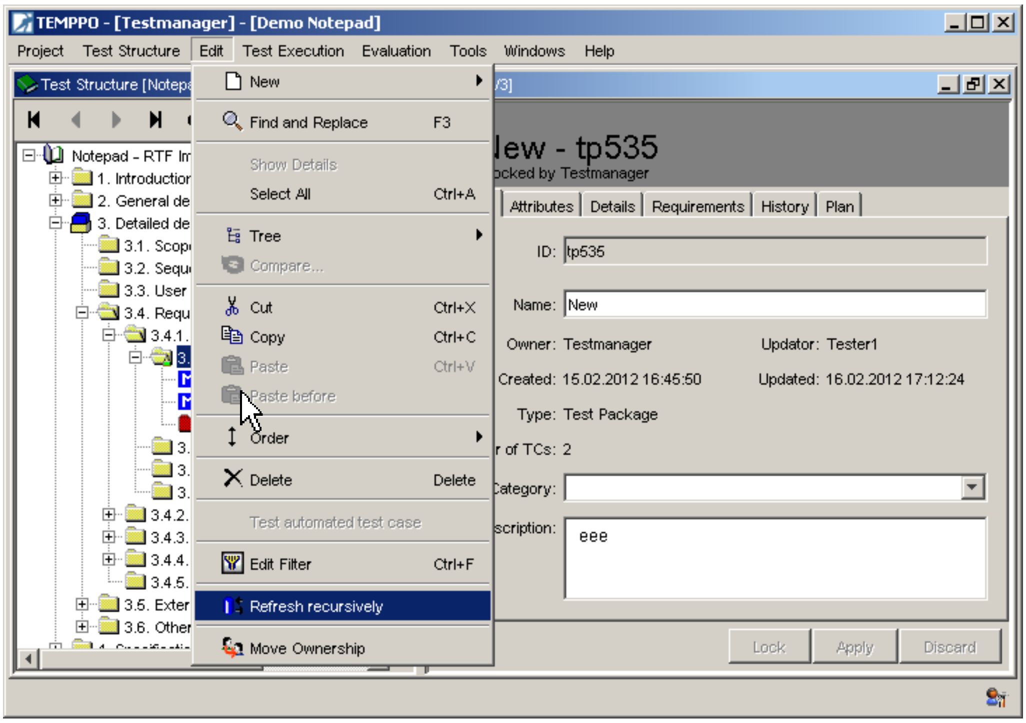
Attribute Values
- Milestone - M1
- Platform - Windows 7 64 bit
- Regressiontest - yes
- Test location - Offshore

Figure 166 - Button Reload for reloading the whole tree

3.2.2.1.2 Refreshing recursively

You can refresh an item including all sub-items by selecting the “refresh recursive” function in the context menu (see Figure 167).

Attention: The selected item with all sub-items will be completely (re)loaded from the database, so it may take a long time.



ID: tp535

Name: New

Owner: Testmanager Updator: Tester1

Created: 15.02.2012 16:45:50 Updated: 16.02.2012 17:12:24

Type: Test Package

r of TCs: 2

Category:

scription: eee

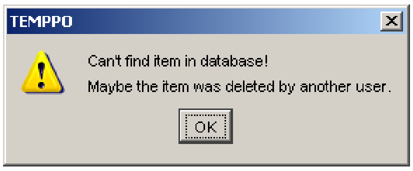
Figure 167 - Refresh recursive menu (also in the context menu)

3.2.2.1.3 Refreshing on certain user actions

An item is automatically refreshed (including its lock state and the presence of all direct children) after the following actions:

- selecting in the tree view
- pressing the **Lock** button

If these actions are done on an actually deleted item, the following dialog is shown and the item is removed from the view.



Can't find item in database!
Maybe the item was deleted by another user.

Figure 168 - Deleted item information (after trying to select or to lock)

3.3 General principles and user prompting

3.3.1 Multi-selection Mode

TEMPPO offers a feature for multi select test packages and test cases in test structures and test suites.

The multi-selection mode can be used for:

- **Assigning attributes / requirements in the test structure**
- **Assigning test result attributes in the test suite**
- **Moving ownership in the test structure**
- **Deleting in the test structure**

- **Setting test results in the test suite**

When the multi-selection mode is activated the items aren't locked automatically. They are only locked during the time for saving. If items are locked by other users the changes are not applied to these items. They are displayed in a list.

3.3.1.1 Multi-selection

The multi-selection mode is activated when more than one item is selected.

If a user selects at least 2 items, the right side (=detailed view) is completely empty.

ⓘ **The user can only activate the multi-selection mode when he has the right.**

In an opened test structure test packages and test cases can be selected.

In a test suite only the test case can be selected. Multi-selection for test packages is not possible.

A multi-selection can be done by using different keys:

- **Select all**

For selecting all, press <Ctrl> and <A>. The user can also activate the context menu **Select All** (Figure 169) or the menu item **Edit → Select All.**

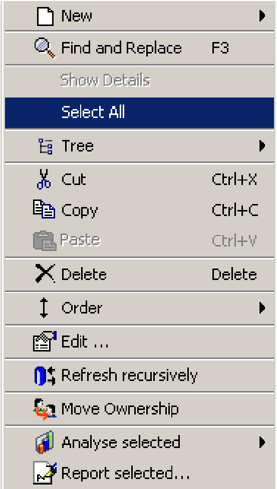
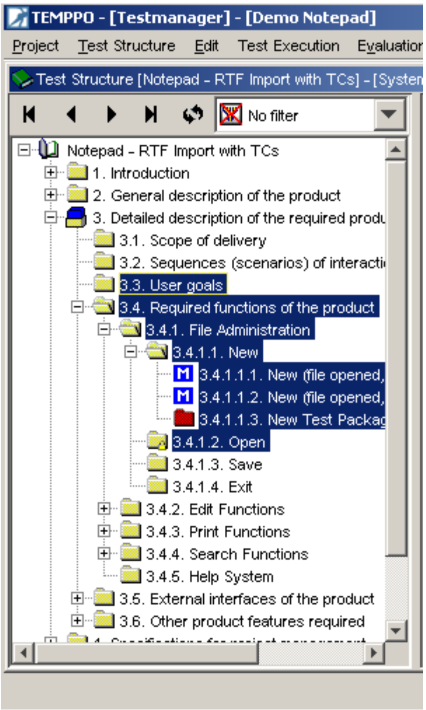


Figure 169 – context menu – "Select All"

The whole sub tree of the selected test structure or test package(s) can be selected (see Figure 170).

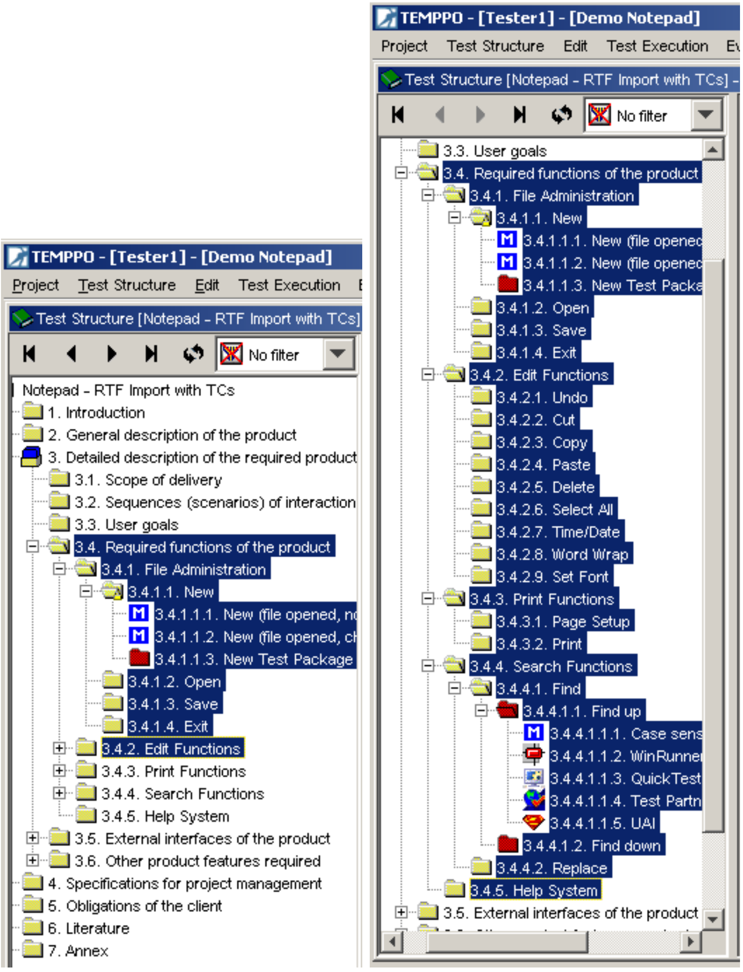


TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Test Structure [Notepad - RTF Import with TCs] - [System
No filter

Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required produ
3.1. Scope of delivery
3.2. Sequences (scenarios) of interacti
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened,
3.4.1.1.2. New (file opened,
3.4.1.1.3. New Test Packag
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.5. Help System
3.5. External interfaces of the product
3.6. Other product features required

Figure 170 – Selection with <Ctrl> + <A> or menu “Select All”

If a test package without children is selected when pressing **Select all** then nothing changes.
Test packages and test cases can be selected with <Ctrl> (see Figure 172) or with <Shift> (see Figure 173). After pressing **Select all** sub trees of the test packages and the selected test cases are marked (see example at Figure 171)



TEMPPO - [Tester1] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Test Structure [Notepad - RTF Import with TCs]
No filter

Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required product
3.1. Scope of delivery
3.2. Sequences (scenarios) of interaction
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no
3.4.1.1.2. New (file opened, ch
3.4.1.1.3. New Test Package
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.5. Help System
3.5. External interfaces of the product
3.6. Other product features required
4. Specifications for project management
5. Obligations of the client
6. Literature
7. Annex

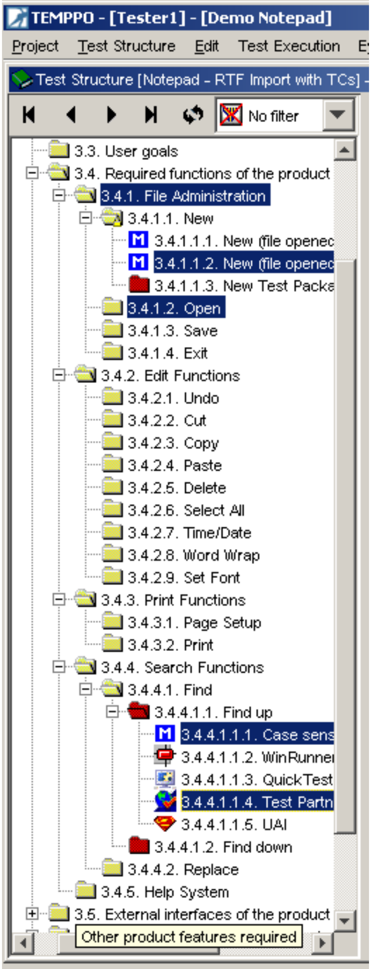
TEMPPO - [Tester1] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Test Structure [Notepad - RTF Import with TCs] -
No filter

3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened
3.4.1.1.2. New (file opened
3.4.1.1.3. New Test Packa
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.2.1. Undo
3.4.2.2. Cut
3.4.2.3. Copy
3.4.2.4. Paste
3.4.2.5. Delete
3.4.2.6. Select All
3.4.2.7. Time/Date
3.4.2.8. Word Wrap
3.4.2.9. Set Font
3.4.3. Print Functions
3.4.3.1. Page Setup
3.4.3.2. Print
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case sens
3.4.4.1.1.2. WinRunner
3.4.4.1.1.3. QuickTest
3.4.4.1.1.4. Test Partn
3.4.4.1.1.5. UAI
3.4.4.1.2. Find down
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of the product

Figure 171 – Before and after "Select all"

- Select non contiguous

For selecting non contiguous, press <Ctrl> and select test packages and/or test cases. A test package is selected itself and not its sub tree, i.e. without children (see Figure 172).

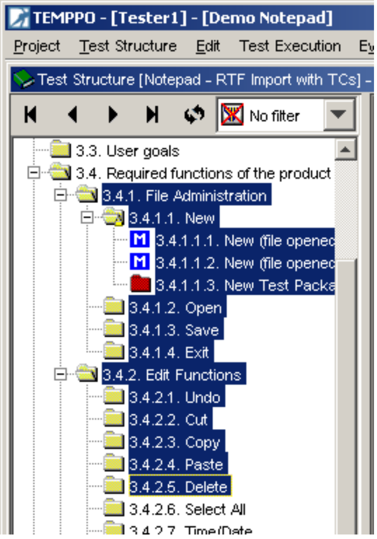

TEMPPO - [Tester1] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Test Structure [Notepad - RTF Import with TCs] -
No filter
Other product features required

3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file openec
3.4.1.1.2. New (file openec
3.4.1.1.3. New Test Packa
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.2.1. Undo
3.4.2.2. Cut
3.4.2.3. Copy
3.4.2.4. Paste
3.4.2.5. Delete
3.4.2.6. Select All
3.4.2.7. Time/Date
3.4.2.8. Word Wrap
3.4.2.9. Set Font
3.4.3. Print Functions
3.4.3.1. Page Setup
3.4.3.2. Print
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Case sens
3.4.4.1.1.2. WinRunner
3.4.4.1.1.3. QuickTest
3.4.4.1.1.4. Test Partn
3.4.4.1.1.5. UAI
3.4.4.1.2. Find down
3.4.4.2. Replace
3.4.5. Help System
3.5. External interfaces of the product

Figure 172 – Selection with <Ctrl>

- Select contiguous

First, select a test package or test case, then press <Shift> and select a test package or test case. The items from the first selected item to the last item are selected, but only the visible items are selected, see Figure 173.


TEMPPO - [Tester1] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Test Structure [Notepad - RTF Import with TCs] -
No filter

3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file openec
3.4.1.1.2. New (file openec
3.4.1.1.3. New Test Packa
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
3.4.2. Edit Functions
3.4.2.1. Undo
3.4.2.2. Cut
3.4.2.3. Copy
3.4.2.4. Paste
3.4.2.5. Delete
3.4.2.6. Select All
3.4.2.7. Time/Date

Figure 173 – Selection with <Shift>

With pressing <Shift> and <Alt> the not visible items are selected, too. The tree is expanded and the items are selected, see and Figure 174.

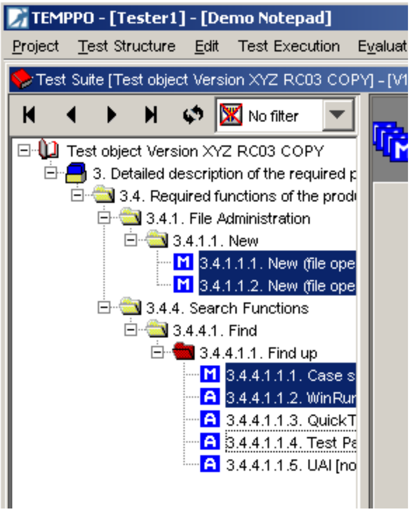


Figure 174 – Tree after selection with <Shift> + <Alt> (in Test Suite)

There are two ways for deselecting:

- Deselect all

Click on any item and the right side changes to the detailed view of this item.

- Deselect single item

Press <**Ctrl**> and click on a selected test package or test case.

If the user has finished his selection, he activates the context menu item **Show Details** (see Figure 175) and the tabs with the content of the selected items is shown on the right side (see Figure 176).

ⓘ **The TEMPPO Designer Task Package and test cases generated by TEMPPO Designer (IDATG) cannot be changed and therefore also not in multi selection mode. If such an artifact is selected, the menu item Show Details is deactivated!**

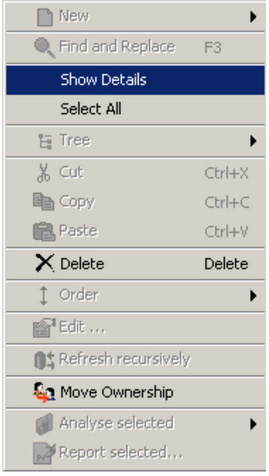
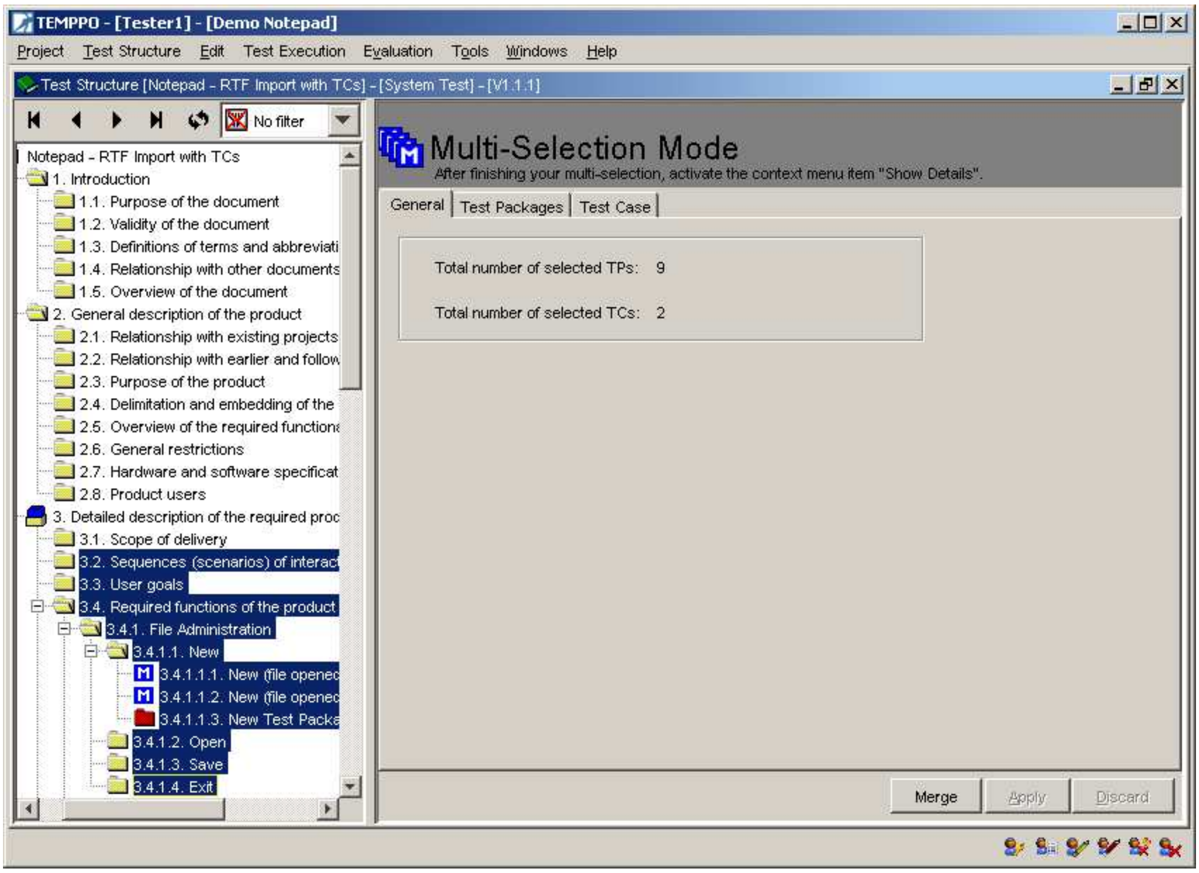


Figure 175 – Show Details


TEMPPO - [Tester1] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1.1.1]
No filter
Notepad - RTF Import with TCs
1. Introduction
1.1. Purpose of the document
1.2. Validity of the document
1.3. Definitions of terms and abbreviati
1.4. Relationship with other documents
1.5. Overview of the document
2. General description of the product
2.1. Relationship with existing projects
2.2. Relationship with earlier and follow
2.3. Purpose of the product
2.4. Delimitation and embedding of the
2.5. Overview of the required functions
2.6. General restrictions
2.7. Hardware and software specificat
2.8. Product users
3. Detailed description of the required proc
3.1. Scope of delivery
3.2. Sequences (scenarios) of interact
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file openec
3.4.1.1.2. New (file openec
3.4.1.1.3. New Test Packa
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
General
Test Packages
Test Case
Merge
Apply
Discard

Multi-Selection Mode

After finishing your multi-selection, activate the context menu item "Show Details".

Total number of selected TPs: 9

Total number of selected TCs: 2

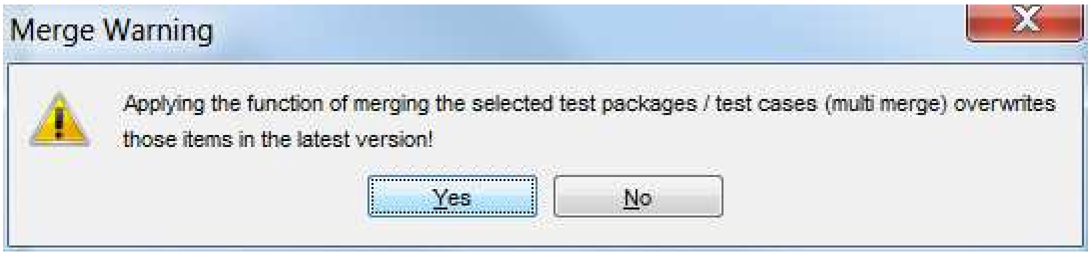
Figure 176 - After "Show Details", content is shown

3.3.1.2 Multi selection merge (of unchanged TPs and TCs)

A so called "Multi selection merge" is done based on the current selection of unchanged TPs and TCs.

ⓘ **Please do not mix this feature with the merge of all changed items!**

If you press the button **Merge**, all items are merged to the latest version after pressing the **Yes** in Figure 176.

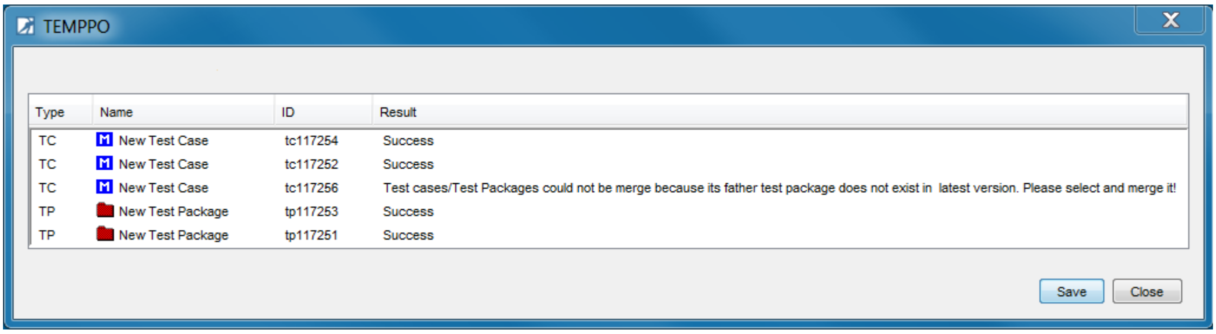

Merge Warning

Applying the function of merging the selected test packages / test cases (multi merge) overwrites those items in the latest version!

Figure 177 – Merge information

After finishing merge a result summary is displayed to the user (see Figure 178).

ⓘ **It is not possible to merge an item, if its father item doesn't exist in the latest version.**



Type	Name	ID	Result
TC	New Test Case	tc117254	Success
TC	New Test Case	tc117252	Success
TC	New Test Case	tc117256	Test cases/Test Packages could not be merge because its father test package does not exist in latest version. Please select and merge it!
TP	New Test Package	tp117253	Success
TP	New Test Package	tp117251	Success

Figure 178 - Merge summary

3.3.1.3 Display of assigned attributes and requirements

3.3.1.3.1 Test Structure

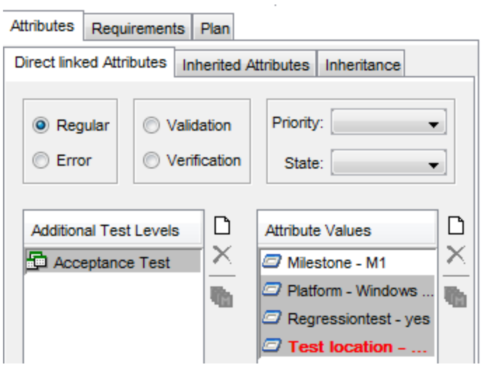
After activating the menu item **Edit > Show Details** two tabs **Test Package** and **Test Case** are displayed. Tab **Test Package** with the sub-tabs **General** and **Requirements** and tab **Test Case** with the sub-tabs **Attributes** and **Requirements** that contain 3 sub-tabs for own attributes/requirements, inherited attributes/requirements and the inheritance state.

If at least two test packages and no test case are selected, the tab **Test Case** is disabled. If at least two test cases and no test package are selected, the tab **Test Package** is disabled.

3.3.1.3.1.1 Test Case, tab Attributes

The tab **Attributes** contains 3 tabs **Direct linked Attributes, Inherited Attributes** and **Inheritance.**

Tab **Direct linked Attributes** displays the attributes which are directly linked to a test case.



● Regular ○ Error ○ Validation ○ Verification
Priority:
State:
Acceptance Test
Milestone - M1
Platform - Windows ...
Regressiontest - yes
Test location - ...

Type and **Situation** (radio buttons):

If all selected test cases have set the same value, the value is selected (●), otherwise there is no selection (○).

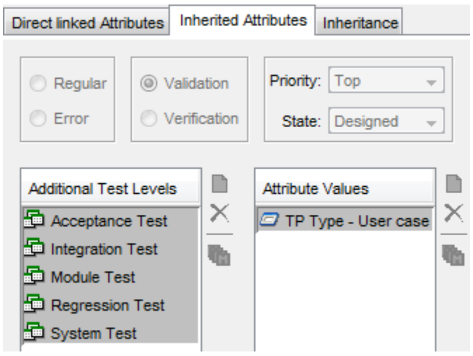
Priority and **State** (combo boxes):

If all selected test cases have set the same value, the value is displayed, otherwise the selection is empty (▾).

Additional Test Levels and **Attribute Values**:

The values that are assigned to all selected test cases are displayed as usual (white background color). The values that are not assigned to all test cases are displayed in a special color (grey background color). If you want to assign such an attribute to all TCs, select it, right mouse click and activate Assign to all test cases. Such attributes or new assigned attributes are displayed in red color.

Tab **Inherited Attributes** displays common state of TCs which have inherited attributes from a TP.



○ Regular ○ Error ● Validation ○ Verification
Priority: Top
State: Designed
Acceptance Test
Integration Test
Module Test
Regression Test
System Test
TP Type - User case

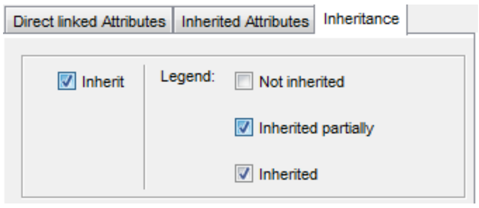
Radio buttons: If all selected test cases with activated inheritance have set the same value, the value is selected (●), otherwise there is no selection (○).

Combo boxes: If all selected test cases with activated inheritance have set the same value, the value is displayed, otherwise the selection is empty (▾).

Additional Test Levels and **Attribute Values**:

The values that are assigned to all selected test cases are displayed as usual (white background color). The values that are not assigned to all test cases are displayed in a special color (grey background color)

The tab **Inheritance** displays the common state of the inheritance flag of the selected TCs. If the checkbox is not activated (□ Not inherited), all selected TCs do not have any inherited attributes. If the checkbox is activated (☑ Inherited), all selected TCs inherit their attributes. If the checkbox looks like ☑ Inherited partially the common state is mixed: TCs with and without inherited attributes.

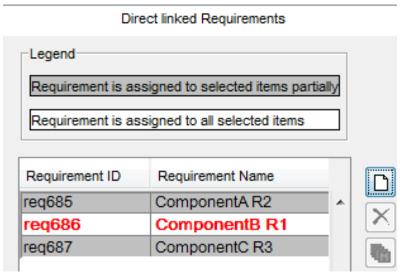


☑ Inherit
Legend: □ Not inherited
☑ Inherited partially
☑ Inherited

3.3.1.3.1.2 Test Case, tab Requirements

The tab **Requirements** contains 3 sub-tabs: **Direct linked Requirements, Inherited Requirements** and **Inheritance**.

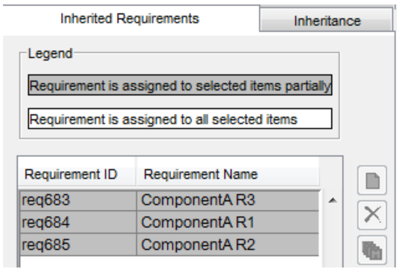
Tab **Direct linked Requirements** displays the requirements which are directly linked to a test case. Requirements assigned to all selected test cases are displayed as usual in white background color. Requirements assigned to not all test cases are displayed in a special color (grey background color). If you want to assign such a requirement to all selected TCs, select it and activate the context menu Assign to all test cases. It is displayed in red color as well as new linked requirements.



Legend
Requirement is assigned to selected items partially
Requirement is assigned to all selected items

Requirement ID	Requirement Name
req685	ComponentA R2
req686	ComponentB R1
req687	ComponentC R3

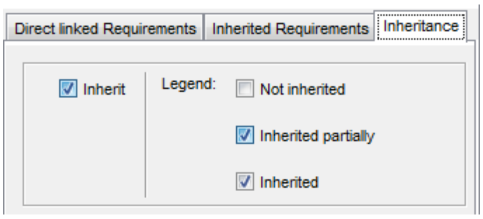
The tab **Inherited Requirements** displays the requirements which are inherited from test packages. Requirements assigned to all selected test cases are displayed as usual in white background color. Requirements assigned to not all test cases are displayed in a special color (grey background color). Of course, it is not possible to edit anything here.



Legend
Requirement is assigned to selected items partially
Requirement is assigned to all selected items

Requirement ID	Requirement Name
req683	ComponentA R3
req684	ComponentA R1
req685	ComponentA R2

The tab **Inheritance** displays the common state of the inheritance flag of the selected TCs. If the checkbox is not activated (☐ Not inherited), all selected TCs do not have any inherited requirements. If the checkbox is activated (☑ Inherited), all selected TCs have at least one inherited requirement. If the checkbox looks like ☑ Inherited partially the common state is mixed: TCs with and without inherited requirements.

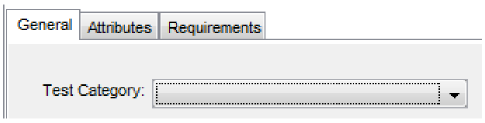


☑ Inherit

Legend: ☐ Not inherited
☑ Inherited partially
☑ Inherited

3.3.1.3.1.3 Test Package, tab General

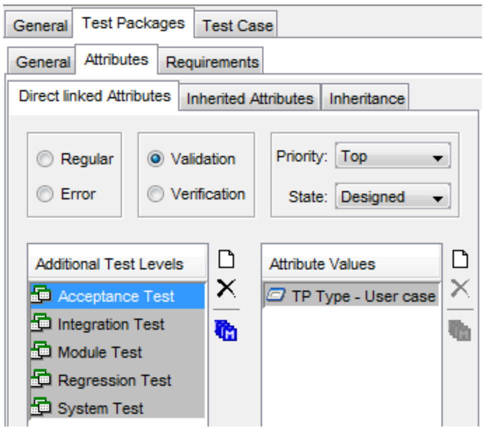
This tab is reduced to the combo box for **test category**. If all selected test packages have set the same value, the value is displayed. If there are test packages with different test categories, an entry with **Several Categories** is selected. If there is no selection, an empty entry is selected.



Test Category:

3.3.1.3.1.4 Test Package, tab Attributes

The tab **Attributes** contains 3 tabs **Direct linked Attributes, Inherited Attributes** and **Inheritance.**



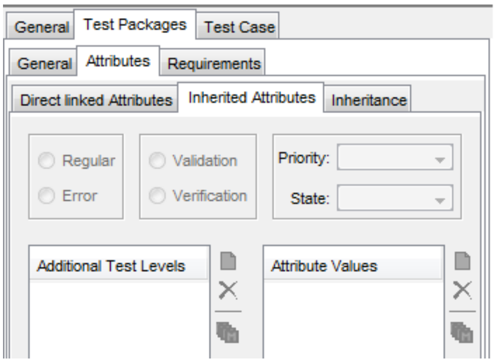
Tab **Direct linked Attributes** displays the common state of all selected TPs that have not set the inherit flag.

Type and **Situation** (radio buttons): If all selected TPs without inherit flag have set the same value, the value is selected (●), otherwise there is no selection (○).

Priority and **State** (combo boxes): If all selected TPs without inherit flag have set the same value, the value is displayed, otherwise the selection is empty (▾).

Additional Test Levels and **Attribute Values**:

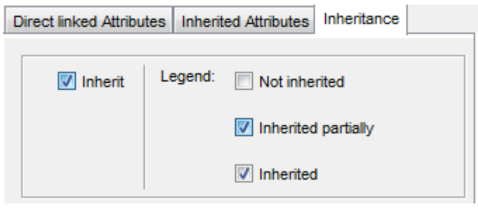
The values that are assigned to all selected test TPs are displayed as usual (white background color). The values that are not assigned to all TPs are displayed in a special color (grey background color). If you want to assign such an attribute to all TPs, select it, right mouse click and activate Assign to all test packages. Such attributes or new assigned attributes are displayed in red color.



Tab **Inherited Attributes** displays common state of TPs which have inherited attributes from a TP.

Radio buttons: If all selected TPs with activated inherit flag have set the same value, the value is selected (●), otherwise there is no selection (○).

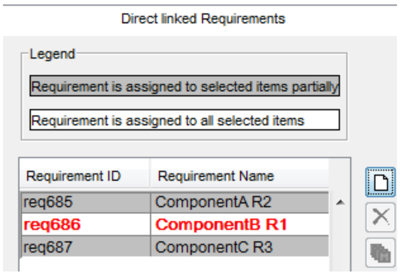
Combo boxes: If all selected TPs with activated inherit flag have set the same value, the value is displayed, otherwise the selection is empty (▾).



The tab **Inheritance** displays the common state of the inheritance flag of the selected TPs. If the checkbox is not activated (□ Not inherited), all selected TPs do not inherit any attributes. If the checkbox is activated (☑ Inherited), all selected TPs inherit their attributes. If the checkbox looks like ☑ Inherited partially the common state is mixed: TPs with and without attribute inheritance.

3.3.1.3.1.5 Test Package, tab Requirements

The tab **Requirements** contains 3 sub-tabs: **Direct linked Requirements, Inherited Requirements** and **Inheritance**.

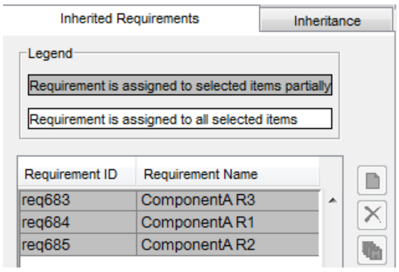


Direct linked Requirements

Legend
Requirement is assigned to selected items partially
Requirement is assigned to all selected items

Requirement ID	Requirement Name
req685	ComponentA R2
req686	ComponentB R1
req687	ComponentC R3

Tab **Direct linked Requirements** displays the requirements which are directly linked to a TP. Requirements assigned to all selected TPs are displayed as usual in white background color. Requirements assigned to not all TPs are displayed in a special color (grey background color). New requirements are displayed in red color. If you want to assign such a requirement to all selected TPs, select it and activate the context menu Assign to all test packages. Such requirements or new linked requirements are displayed in red color.

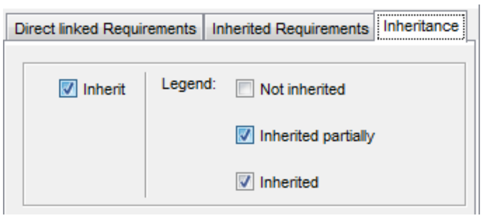


Inherited Requirements | Inheritance

Legend
Requirement is assigned to selected items partially
Requirement is assigned to all selected items

Requirement ID	Requirement Name
req683	ComponentA R3
req684	ComponentA R1
req685	ComponentA R2

The tab **Inherited Requirements** displays the common state of selected TPs which inherit their requirements. Requirements assigned to all selected TPs are displayed as usual in white background color. Requirements assigned to not all TPs are displayed in a special color (grey background color). Of course, it is not possible to edit anything here.



Direct linked Requirements | Inherited Requirements | Inheritance

☑ Inherit

Legend:
☐ Not inherited
☑ Inherited partially
☑ Inherited

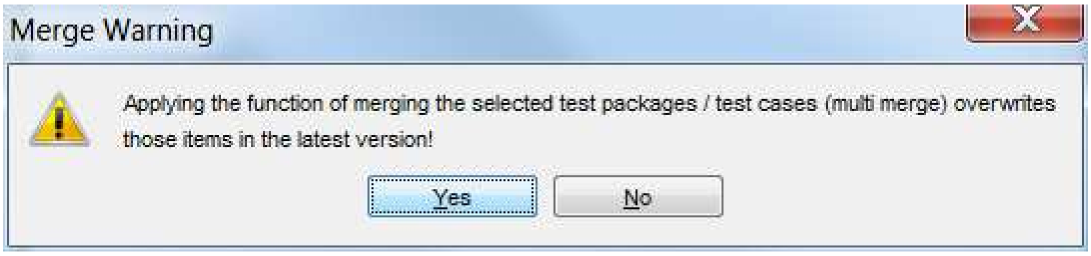
The tab **Inheritance** displays the common state of the inheritance flag of the selected TPs. If the checkbox is not activated (☐ Not inherited), all selected TPs do not have any inherited requirements. If the checkbox is activated (☑ Inherited), all selected TPs have at least one inherited requirement. If the checkbox looks like ☑ Inherited partially the common state is mixed: TPs with and without inherited requirements.

3.3.1.4 Multi selection merge (of changed TPs and TCs)

A so called "Multi selection merge (of changed TPs and TCs)" is done based on the changed TPs and TCs of the current selection. Only the changed ones are merged!

ⓘ **Please do not mix this feature with the merge of all items!**

If you press the button **Merge**, all items are merged to the latest version after pressing the **Yes** in Figure 179.



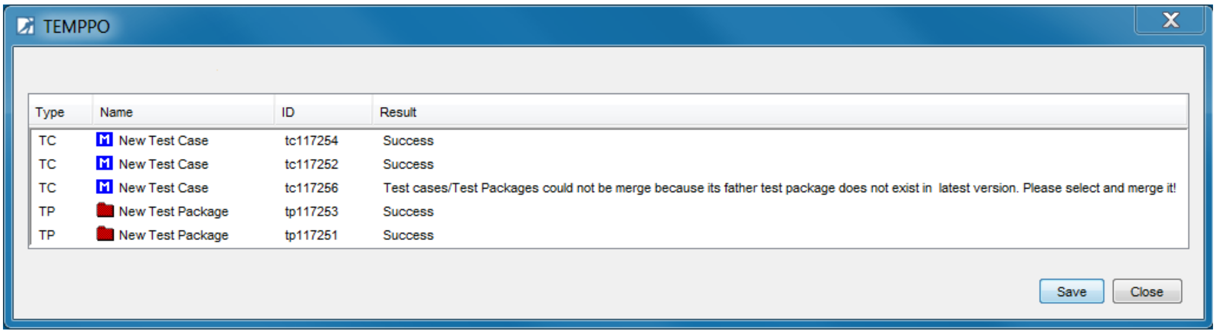
Merge Warning

Applying the function of merging the selected test packages / test cases (multi merge) overwrites those items in the latest version!

Figure 179 – Merge information

After finishing merge a result summary is displayed to the user.

ⓘ **It is not possible to merge an item, if its father item doesn't exist in the latest version.**



TEMPPO

Type	Name	ID	Result
TC	New Test Case	tc117254	Success
TC	New Test Case	tc117252	Success
TC	New Test Case	tc117256	Test cases/Test Packages could not be merge because its father test package does not exist in latest version. Please select and merge it!
TP	New Test Package	tp117253	Success
TP	New Test Package	tp117251	Success

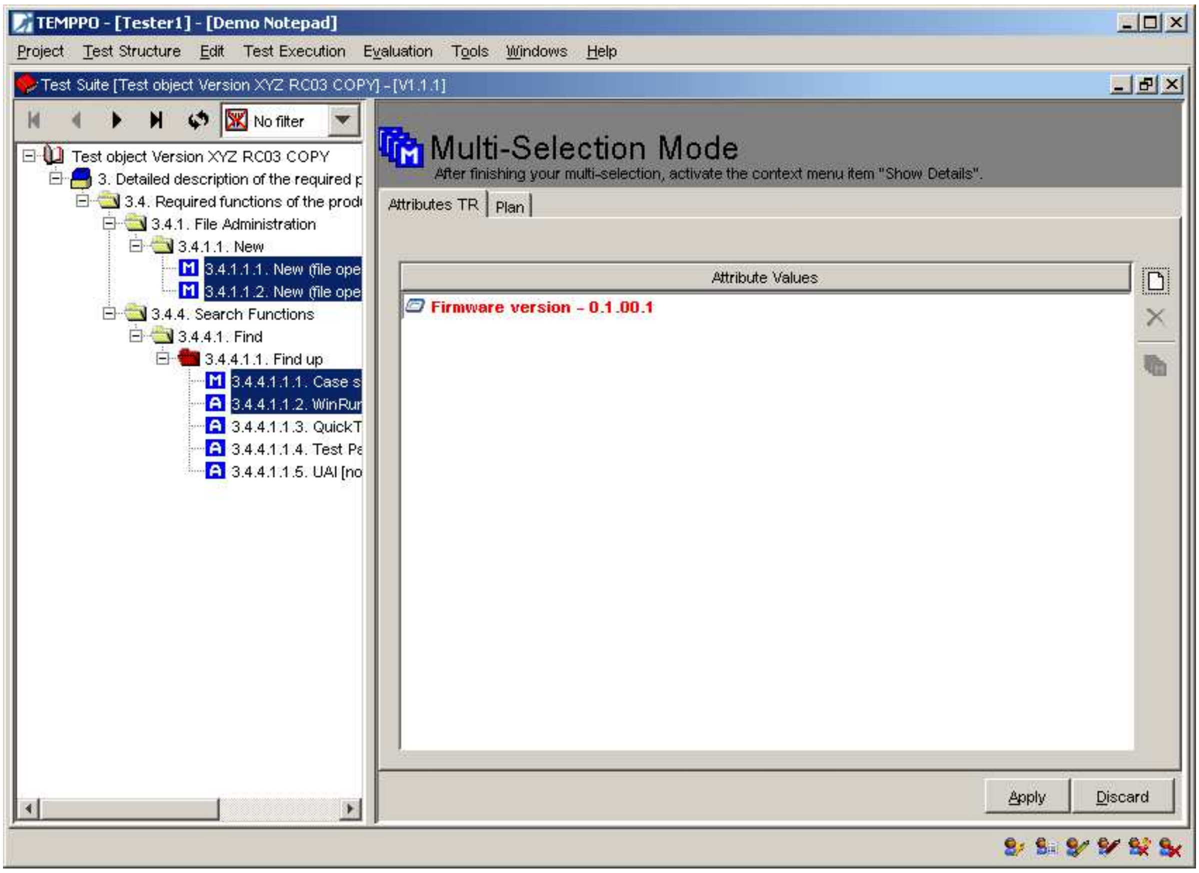
Figure 180 - Merge summary

3.3.1.4.1Test Suite

After activating the menu **Edit > Show Details** the tab Attributes TR is displayed.

- tab **Attributes TR** (see Figure 181)

The attribute values that are assigned to all selected test cases are displayed as usual. The attribute values assigned to not all test cases are displayed in a special color (grey background color). New added attributes are displayed in red color.



After finishing your multi-selection, activate the context menu item "Show Details".

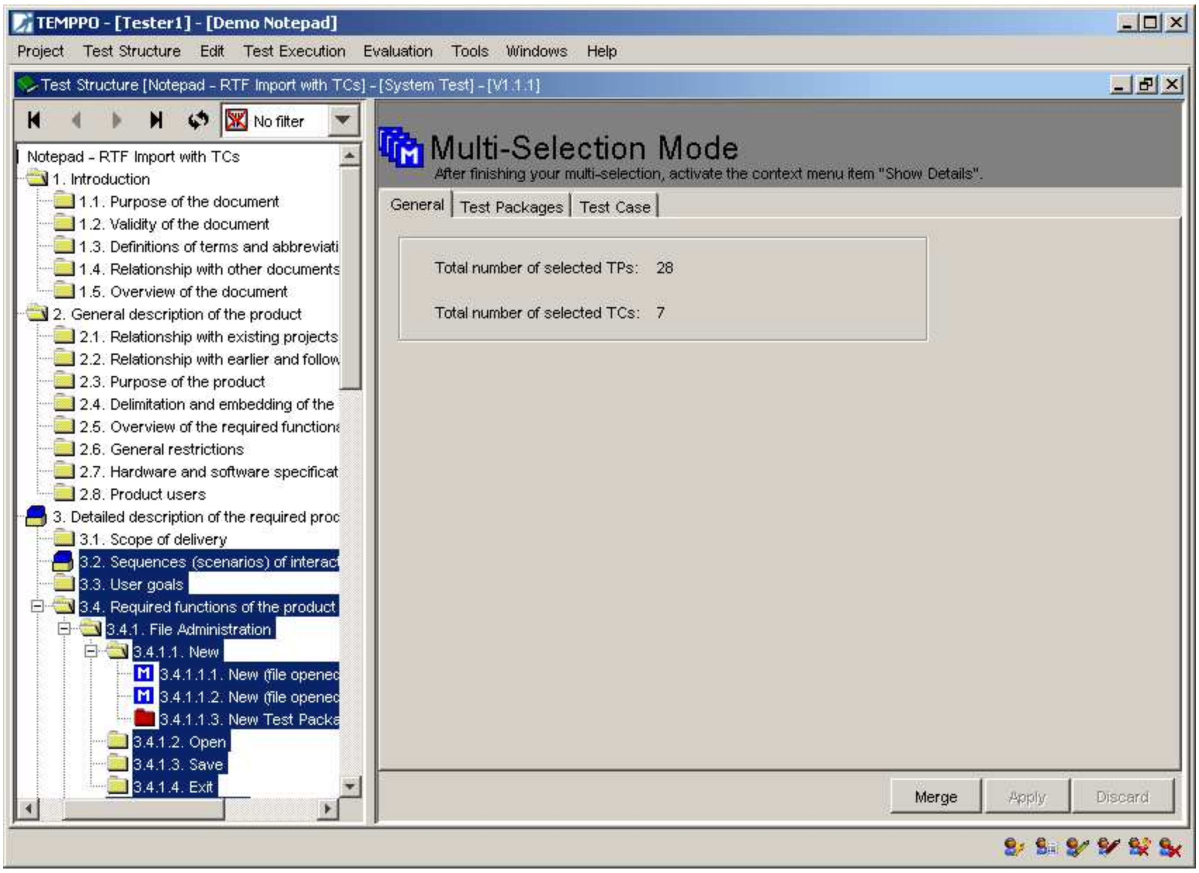
Firmware version - 0.1.00.1

Figure 181 – Attributes TR

3.3.1.4.2Filter in multi-selection mode

If there is a multi-selection and a filter is set, only those items that match are considered. If the items of the multi-selection are still shown, they are selected again (see Figure 182 and Figure 183).

The **Show Details** menu item has to be activated again, because the selection has changed.


TEMPPO - [Tester1] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1.1.1]
No filter
Notepad - RTF Import with TCs
1. Introduction
1.1. Purpose of the document
1.2. Validity of the document
1.3. Definitions of terms and abbreviati
1.4. Relationship with other documents
1.5. Overview of the document
2. General description of the product
2.1. Relationship with existing projects
2.2. Relationship with earlier and follow
2.3. Purpose of the product
2.4. Delimitation and embedding of the
2.5. Overview of the required functiona
2.6. General restrictions
2.7. Hardware and software specificat
2.8. Product users
3. Detailed description of the required proc
3.1. Scope of delivery
3.2. Sequences (scenarios) of interact
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file openec
3.4.1.1.2. New (file openec
3.4.1.1.3. New Test Packa
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
Multi-Selection Mode
After finishing your multi-selection, activate the context menu item "Show Details".
General
Test Packages
Test Case
Merge
Apply
Discard

Total number of selected TPs: 28

Total number of selected TCs: 7

Figure 182 – No filter

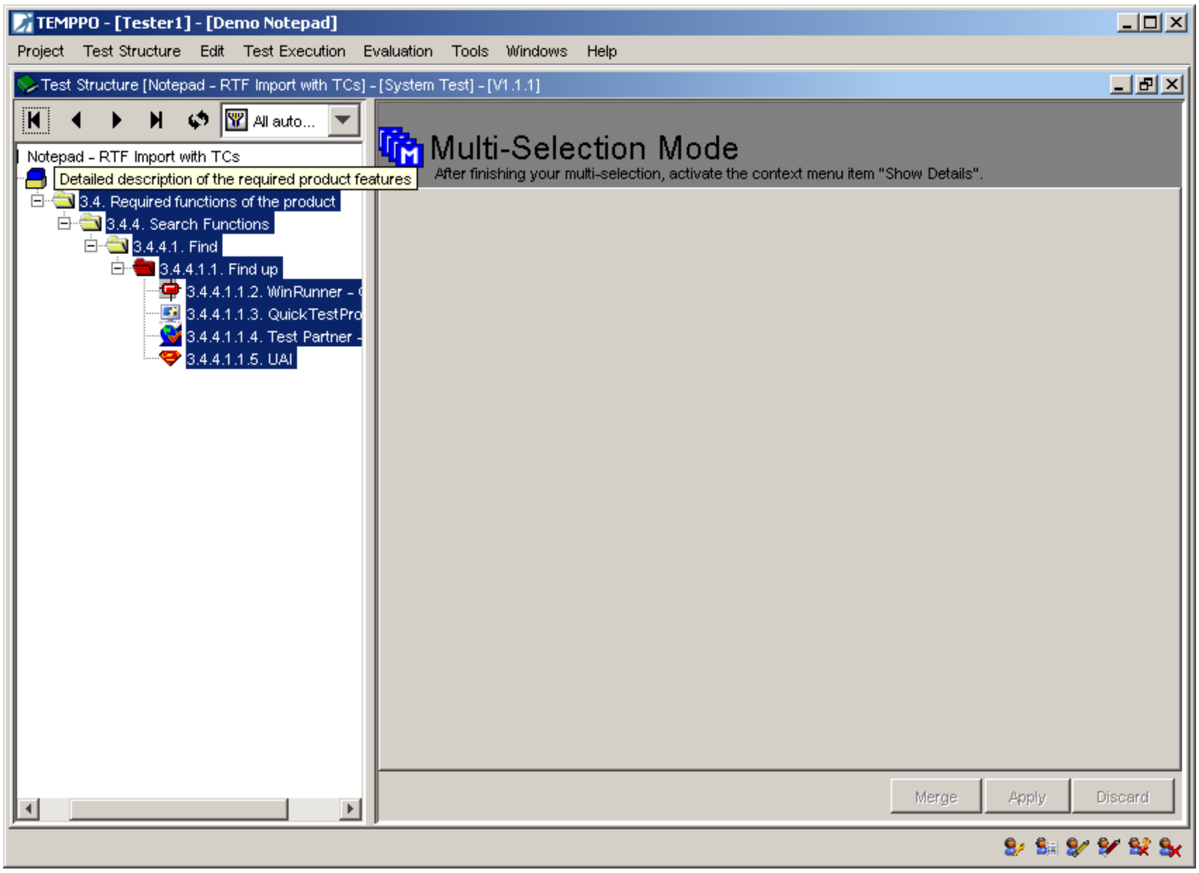

TEMPPO - [Tester1] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V1.1.1]
All auto...
Notepad - RTF Import with TCs
Detailed description of the required product features
3.4. Required functions of the product
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.2. WinRunner -
3.4.4.1.1.3. QuickTestPro
3.4.4.1.1.4. Test Partner -
3.4.4.1.1.5. UAI
Multi-Selection Mode
After finishing your multi-selection, activate the context menu item "Show Details".
Merge
Apply
Discard

Figure 183 – Set Filter

3.3.1.5 Setting new assignments

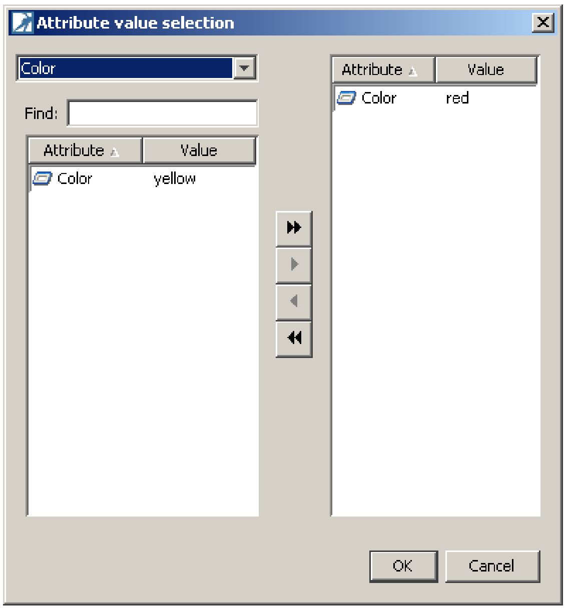
The user can set or remove attributes values or requirements to / from the selected items. If any changes are made, the buttons **Apply** and **Discard** get enabled.

Values of radio buttons and combo boxes:

If a value is set and the button **Apply** is pressed, all selected items get that value. If no value is set, nothing changes.

Values in lists:

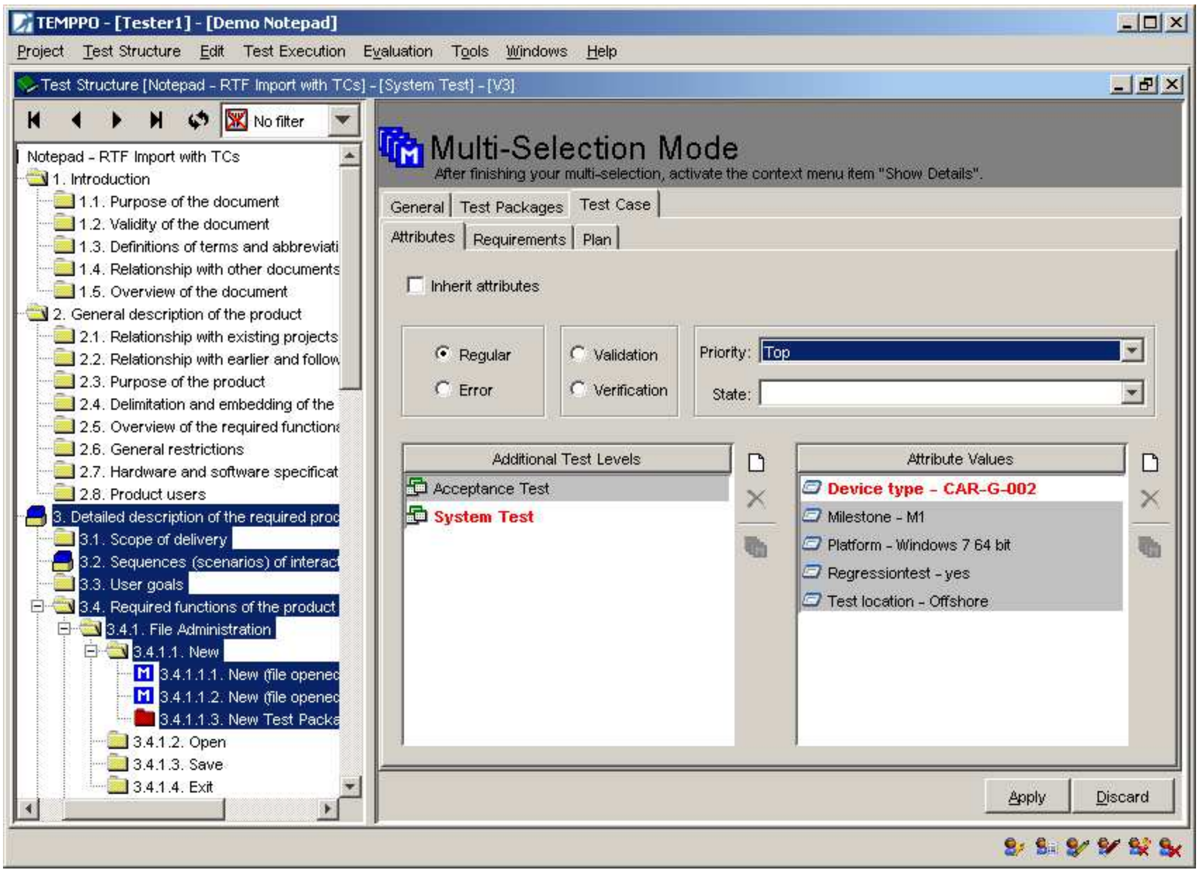
"Common values" can be set and removed. The assignment of additional test levels, attribute values and requirements to the selected items is done analogue to the assignment of these values to a single item with the assignment dialog (see Figure 184). A new common value is written in red with a white background color (see Figure 185).



Attribute	Value
Color	yellow

Attribute	Value
Color	red

Figure 184 – Assignment dialog


TEMPPO - [Tester1] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. Introduction
1.1. Purpose of the document
1.2. Validity of the document
1.3. Definitions of terms and abbreviati
1.4. Relationship with other documents
1.5. Overview of the document
2. General description of the product
2.1. Relationship with existing projects
2.2. Relationship with earlier and follow
2.3. Purpose of the product
2.4. Delimitation and embedding of the
2.5. Overview of the required functiona
2.6. General restrictions
2.7. Hardware and software specificat
2.8. Product users
3. Detailed description of the required proc
3.1. Scope of delivery
3.2. Sequences (scenarios) of interact
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file openec
3.4.1.1.2. New (file openec
3.4.1.1.3. New Test Packa
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
Multi-Selection Mode
General
Test Packages
Test Case
Attributes
Requirements
Plan
Apply
Discard

After finishing your multi-selection, activate the context menu item "Show Details".

☐ Inherit attributes

● Regular ○ Error

○ Validation ○ Verification

Priority: Top

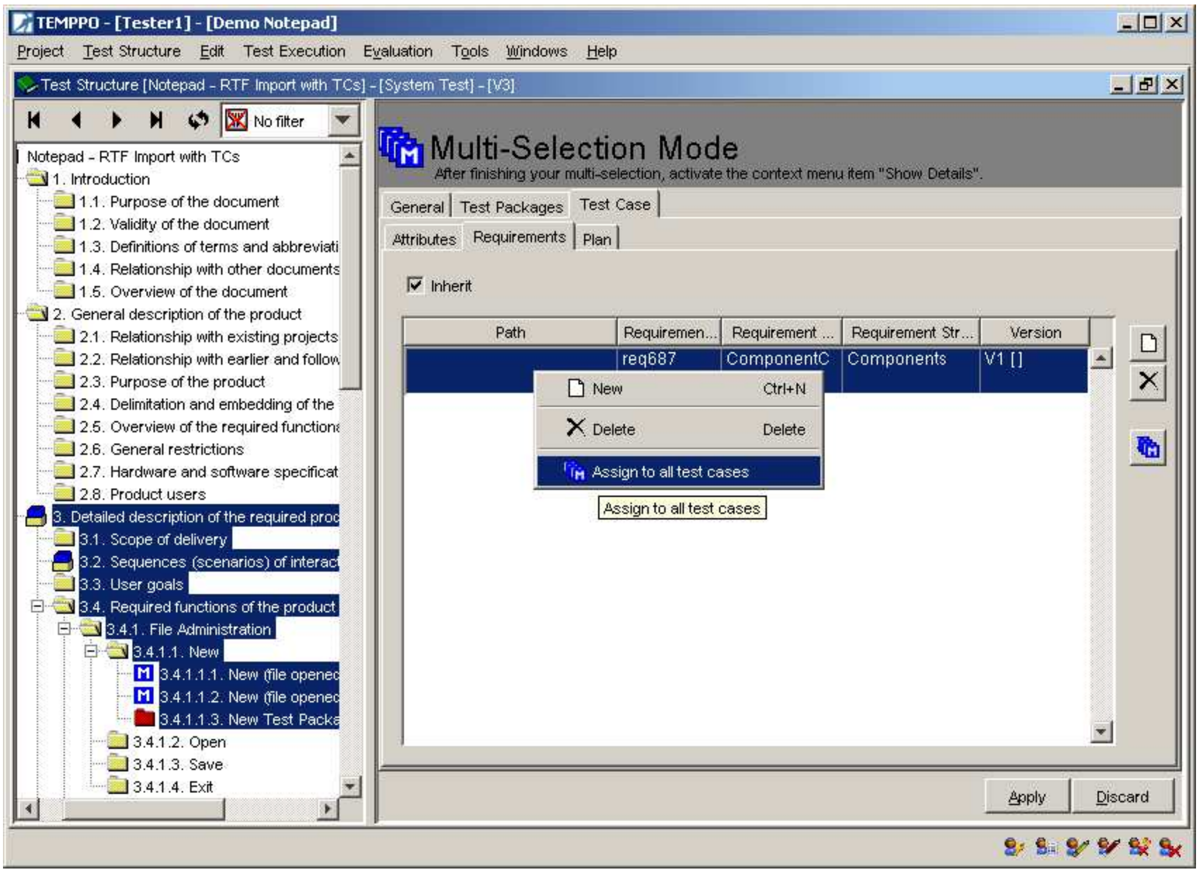
State:

Additional Test Levels
Acceptance Test
System Test

Attribute Values
Device type - CAR-G-002
Milestone - M1
Platform - Windows 7 64 bit
Regressiontest - yes
Test location - Offshore

Figure 185 – New common values

“Not common values” displayed in grey can only be removed. But if a user wants to apply a “grey” value to all selected items, he activates the button **Assign to all test cases.** Then this value is written in red with a grey background color (see Figure 186).

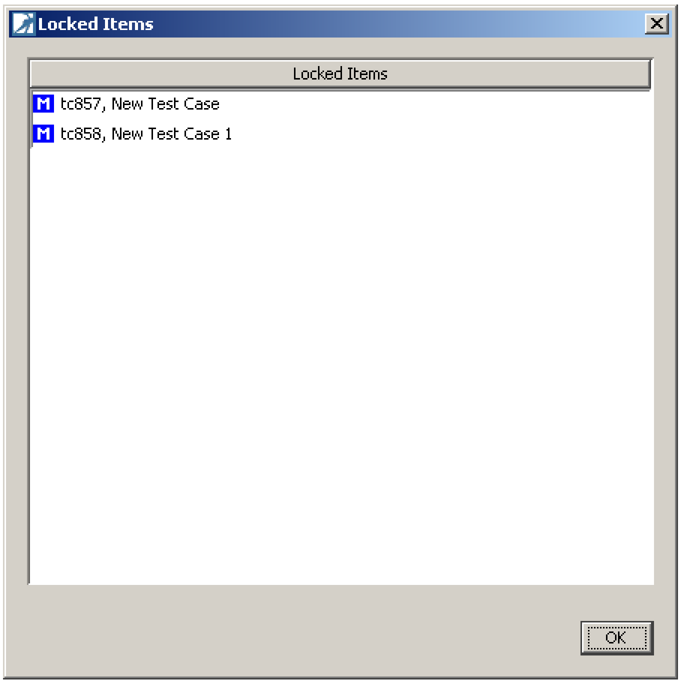

TEMPPO - [Tester1] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. Introduction
1.1. Purpose of the document
1.2. Validity of the document
1.3. Definitions of terms and abbreviati
1.4. Relationship with other documents
1.5. Overview of the document
2. General description of the product
2.1. Relationship with existing projects
2.2. Relationship with earlier and follow
2.3. Purpose of the product
2.4. Delimitation and embedding of the
2.5. Overview of the required functiona
2.6. General restrictions
2.7. Hardware and software specificat
2.8. Product users
3. Detailed description of the required proc
3.1. Scope of delivery
3.2. Sequences (scenarios) of interact
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file openec
3.4.1.1.2. New (file openec
3.4.1.1.3. New Test Packa
3.4.1.2. Open
3.4.1.3. Save
3.4.1.4. Exit
Multi-Selection Mode
After finishing your multi-selection, activate the context menu item "Show Details".
General
Test Packages
Test Case
Attributes
Requirements
Plan
☑ Inherit
New Ctrl+N
Delete Delete
Assign to all test cases
Assign to all test cases
Apply
Discard

Path	Requiremen...	Requirement ...	Requirement Str...	Version
	req687	ComponentC	Components	V1 []

Figure 186 – Assign to all test cases

If the button **Apply** is pressed, all common values (white background color and red foreground color) are set for the selected items. All values that have been removed on the GUI will be removed from the selected items.

Changes are not applied to locked items, which are displayed in a list (see Figure 187).



Locked Items
tc857, New Test Case
tc858, New Test Case 1

Figure 187 – Locked Items…

If the button **Discard** is pressed, all changes in all tabs will be discarded.

3.3.1.6 Move Ownership

Now the function “move ownership” can be applied for any number of selected test cases and test packages, but the right has to be considered.

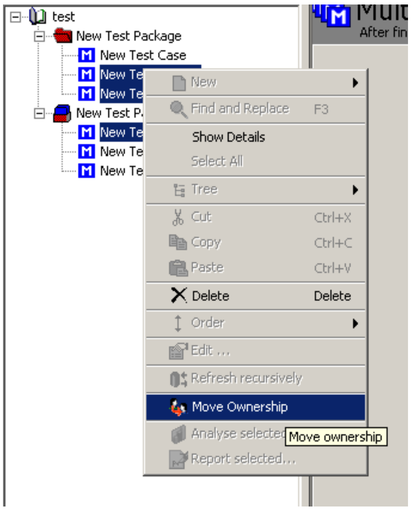
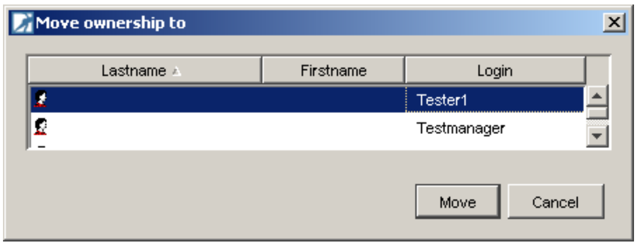


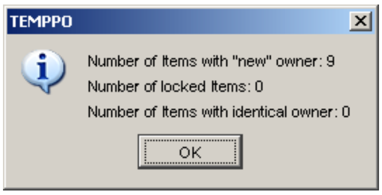
Figure 188 – Context menu – Move Ownership


Move ownership to

Lastname	Firstname	Login
		Tester1
		Testmanager

Figure 189 – Move ownership

After the process moving the ownership, a summary is shown (see Figure 190).


TEMPPO

Number of Items with "new" owner: 9
Number of locked Items: 0
Number of Items with identical owner: 0

Figure 190 - Summary

If the user doesn’t have the right to move the ownership of foreign items or the item is locked, a list of items is displayed (see Figure 191). These items aren’t changed.

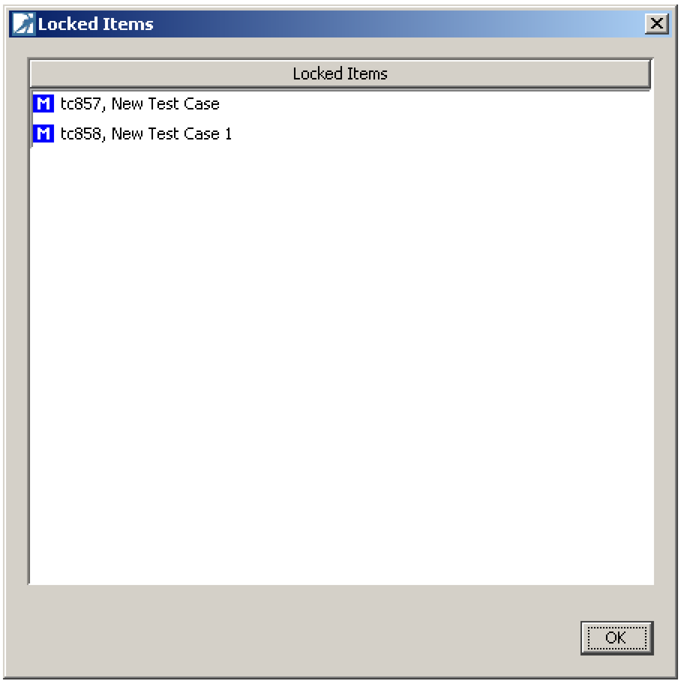


Figure 191 – Locked Items…

3.3.1.7 Delete

This function is now available for any number of selected test cases or test packages, but the user has to have the right.

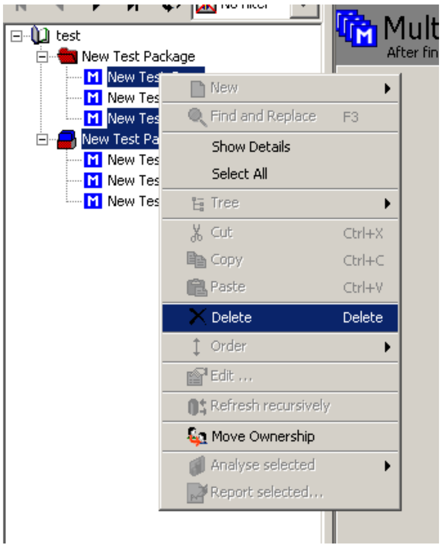


Figure 192 – Context menu - delete

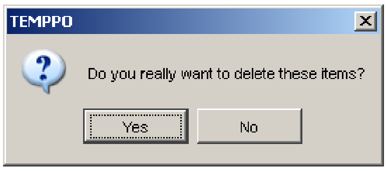
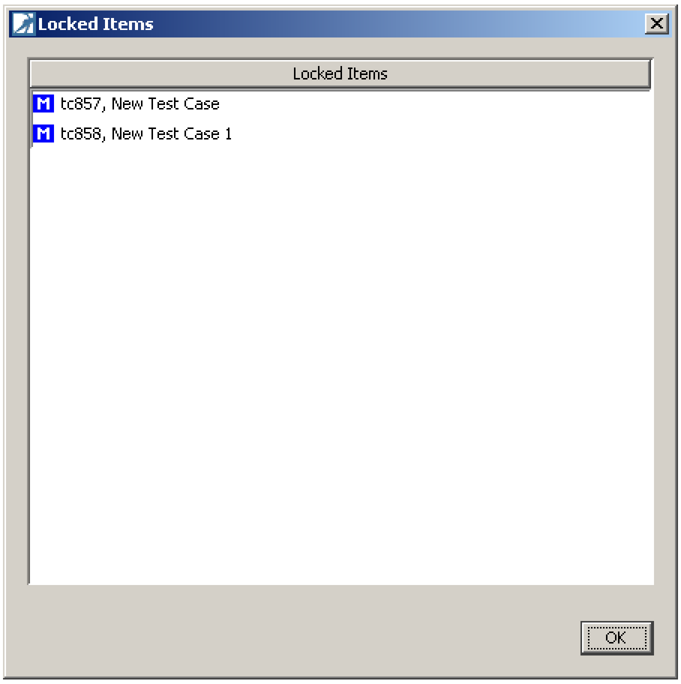


Figure 193 – Acknowledgement for deleting

After the acknowledgement of the user the delete process starts. If the items are locked or the user doesn't have the right to delete "foreign" items, a list of these items which couldn't be removed is displayed (see Figure 194).



tc857, New Test Case

tc858, New Test Case 1

Figure 194 – Locked Items…

3.3.1.8 Set test results

The function "**Set Result(s)**" can be applied either for a whole sub tree or for any number of selected test cases. Activate **Set Result(s)** in the context menu (see Figure 195) and afterwards Figure 196 is shown.

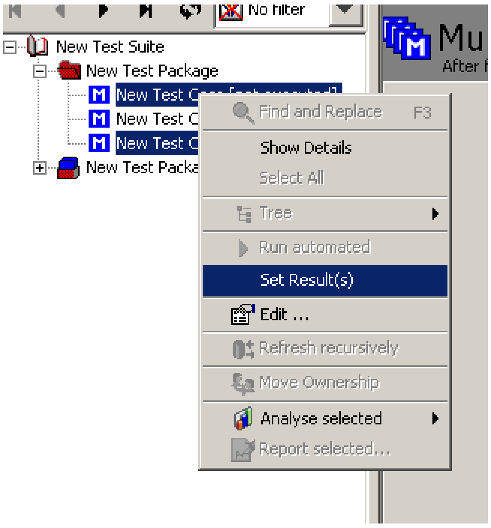
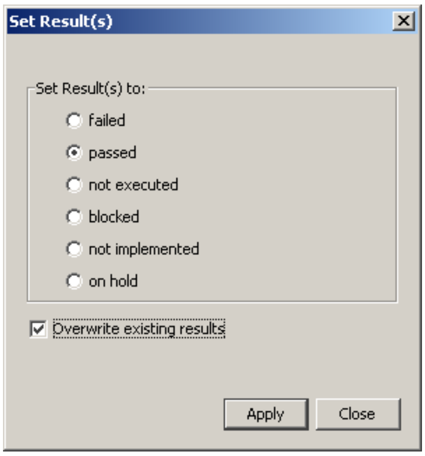


Figure 195 – Context menu – Set Result(s)



Set Result(s)

Set Result(s) to:

○ failed
● passed
○ not executed
○ blocked
○ not implemented
○ on hold

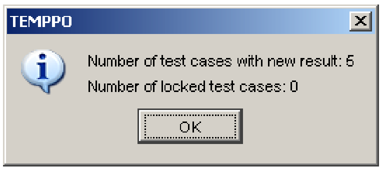
☑ Overwrite existing results

Figure 196 – Set Result(s)

You can choose one result and optionally set the checkbox **Overwrite existing results** and after clicking **Apply** all test results are set.

ⓘ **If other users have locked test cases, it is not possible to set the result, of course. You will be informed if results couldn't be set (see Figure 197).**

ⓘ **If a test case contains no test steps, it remains "not executed".**



TEMPPO

Number of test cases with new result: 5
Number of locked test cases: 0

Figure 197 – Information message

3.3.2 Hyperlinks and Uploads

Generally you can mark text as

- **Hyperlink to the web**
- **Hyperlink to a file**
- **Hyperlink to an upload**
- **Hyperlink to a test automation tool or command**
- **Hyperlink to a test case**

3.3.2.1 Creating hyperlinks

Hyperlinks to the web, to the file system, to uploads, to a test automation tool, to a command or to a test case can be created for the following text fields

- **Description** (project, test structure, test package)
- **Precondition** (test package, test case)
- **Postcondition** (test package, test case)
- **Test goal** (test case)
- **Instruction, input, expected** (test case)
- **Result output, actual result, bug ID** (test case in test suite)

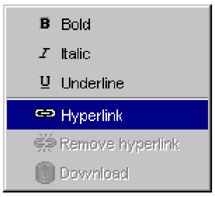
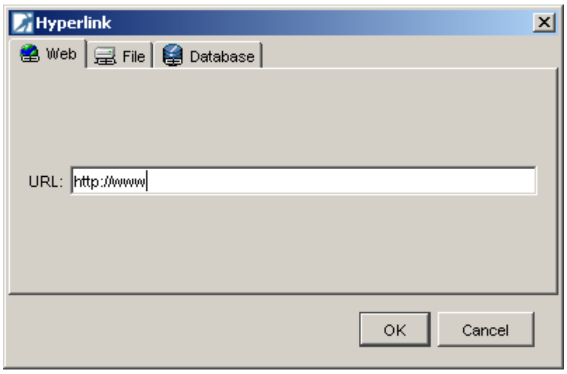


Figure 198 - Menu hyperlink

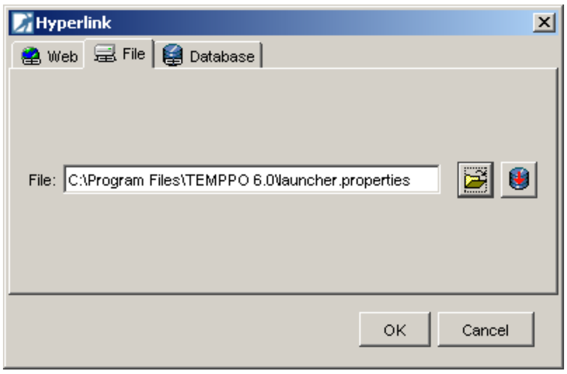
For creating a hyperlink it is necessary to select at least one character in editable text fields and activate the context sensitive menu **Hyperlink** (see Figure 198), which opens the Hyperlink window (see Figure 199).



URL: http://www

Figure 199 - Hyperlink to the web

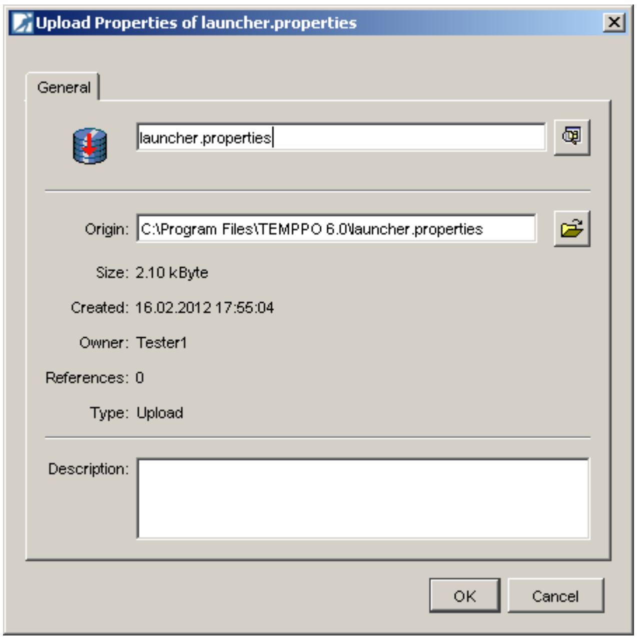
You have the possibility to enter a link to the web (specified by an HTTP address) or to the file system (see Figure 200).



File: C:\Program Files\TEMPPO 6.0\launcher.properties

Figure 200 - Hyperlink to the file system

In case of changing drive letters, moving file shares and other similar problems it may be better to upload the files to the database. For referencing an upload you have 2 possibilities: you can upload a new file or use an existing one and create a reference on it. The philosophy of this concept is to upload a file only once and create more references on it (if necessary).



launcher.properties

Origin: C:\Program Files\TEMPPO 6.0\launcher.properties

Size: 2.10 kByte

Created: 16.02.2012 17:55:04

Owner: Tester1

References: 0

Type: Upload

Description:

Figure 201 - Upload properties

Upload a new file: You have to activate the **Upload** button to open Figure 201 for specifying a special **name** and **description**. Additionally **owner**, **creation date**, **size**, the number of **references** and the **type** are displayed.

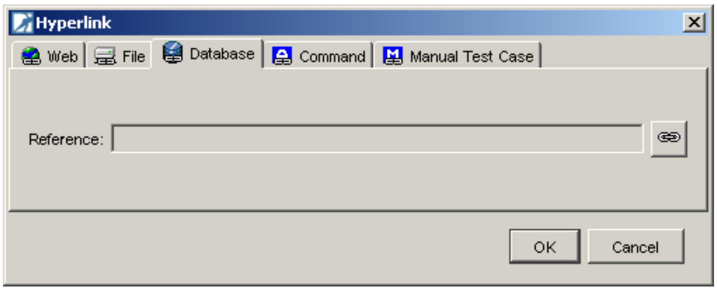
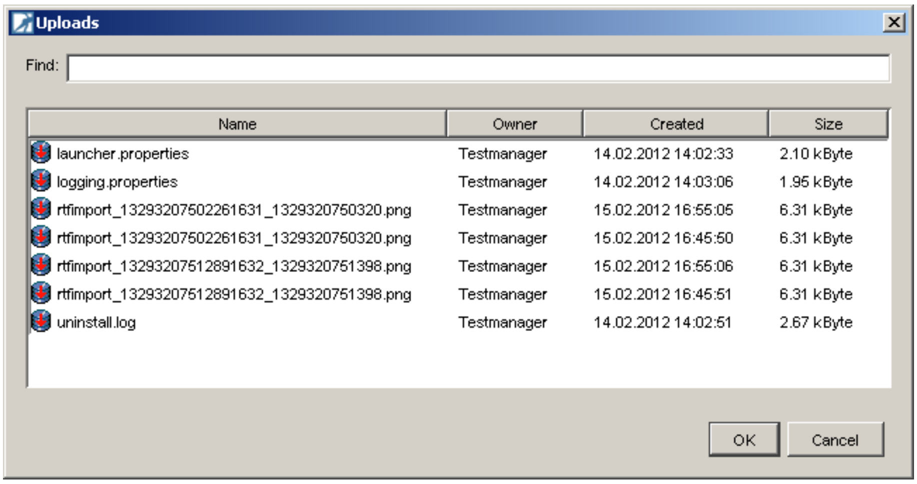


Figure 202 - Hyperlink database

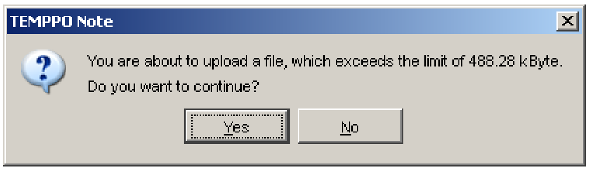
Use an existing upload: You have to activate tab **database** and press button **create reference to uploaded file** to open Figure 203, which shows all uploads of the database. For creating a new reference on an existing upload you simply have to select it and click **OK** in Figure 203 and Figure 202.



Name	Owner	Created	Size
launcher.properties	Testmanager	14.02.2012 14:02:33	2.10 kByte
logging.properties	Testmanager	14.02.2012 14:03:06	1.95 kByte
rtfimport_1329320750226l631_1329320750320.png	Testmanager	15.02.2012 16:55:05	6.31 kByte
rtfimport_1329320750226l631_1329320750320.png	Testmanager	15.02.2012 16:45:50	6.31 kByte
rtfimport_1329320751289l632_1329320751398.png	Testmanager	15.02.2012 16:55:06	6.31 kByte
rtfimport_1329320751289l632_1329320751398.png	Testmanager	15.02.2012 16:45:51	6.31 kByte
uninstall.log	Testmanager	14.02.2012 14:02:51	2.67 kByte

Figure 203 - Uploads

ⓘ **The upload limit of a file is set to 0.5 MB by default and is specified in the file admin.properties (dbsettings.upload.limit=500000). But it is only information. Of course, you can upload files that exceed that limit and/or change this value in admin.properties.**



You are about to upload a file, which exceeds the limit of 488.28 kByte.
Do you want to continue?

Figure 204 – File size exceeds the upload limit

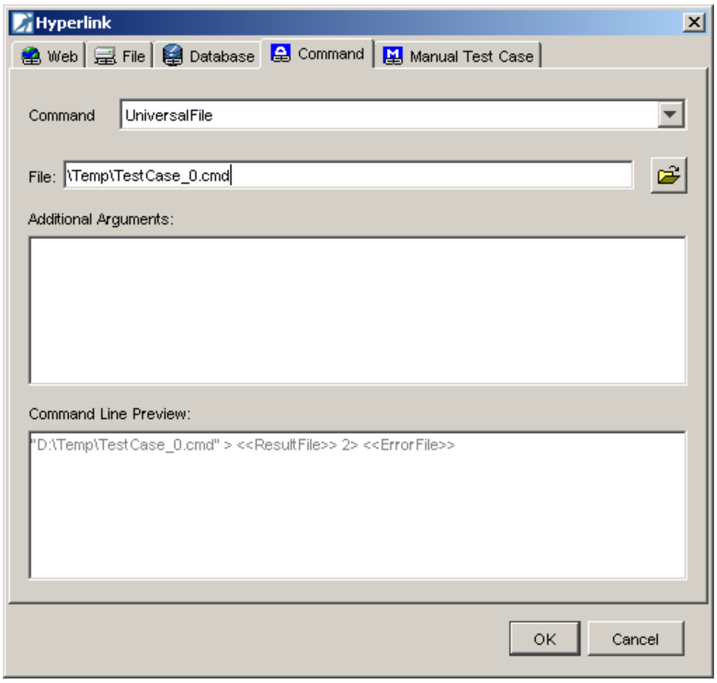
The fourth possibility is that you create hyperlinks with parameters:

- by a test automation tool
- or a command defined in settings (see 3.3.20.2.10)

which is selected in the first combo box.

Hyperlink with a test automation tool are defined like automated test cases. You have to define a file and (optional) additional arguments (for more details see 3.1.3.1.6).

Pressing on the hyperlink defined with a test automation tool, executes the link and then a message box is shown with the result.



Command: UniversalFile

File: \Temp\TestCase_0.cmd

Additional Arguments:

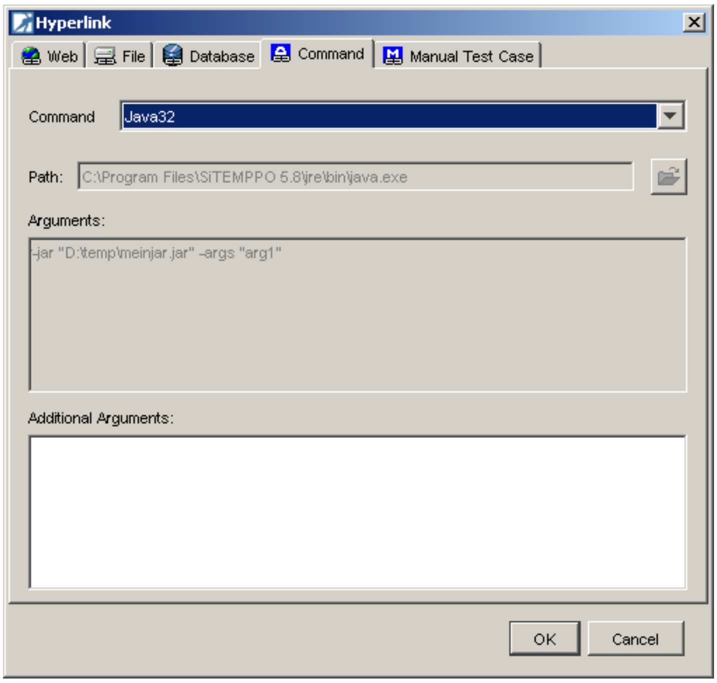
Command Line Preview:

```
"D:\Temp\TestCase_0.cmd" > <<ResultFile>> 2> <<ErrorFile>>
```

Figure 205 – Hyperlink test automation tool

Hyperlink with a command can be defined with additional arguments.

Pressing on the hyperlink defined with a command, then the command is executed and then a message box shows the output of the process.



Command: Java32

Path: C:\Program Files\SiTEMPPO 5.8\jre\bin\java.exe

Arguments:

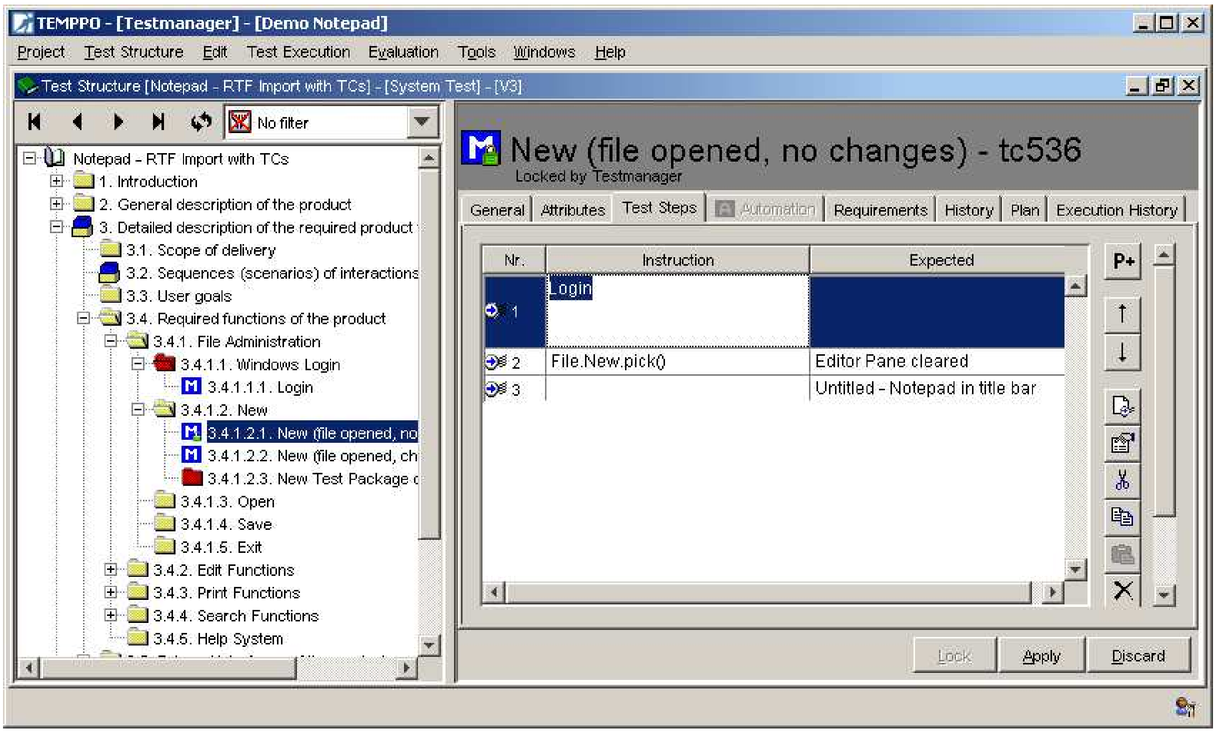
```
-jar "D:\temp\meinjar.jar" -args "arg1"
```

Additional Arguments:

Figure 206 – Hyperlink command

Hyperlink to test case is defined if a test case or test steps are reused in another test case. E.g., if you create a test case “Login” with some steps that is used other test cases, it can be referenced by hyperlinks from other test cases.

First of all you create a step, and select the text that becomes a hyperlink:


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required product
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. Windows Login
3.4.1.1.1. Login
3.4.1.2. New
3.4.1.2.1. New (file opened, no
3.4.1.2.2. New (file opened, ch
3.4.1.2.3. New Test Package c
3.4.1.3. Open
3.4.1.4. Save
3.4.1.5. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.5. Help System
New (file opened, no changes) - tc536
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
P+
Lock Apply Discard

Nr.	Instruction	Expected
1	Login	
2	File.New.pick()	Editor Pane cleared
3		Untitled - Notepad in title bar

Figure 207 - Source test case

Then click right mouse button for context menu, select **Hyperlink**” and select the test case.

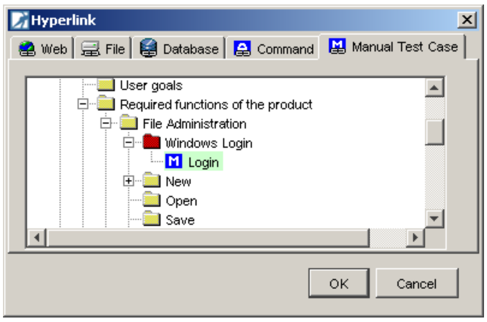
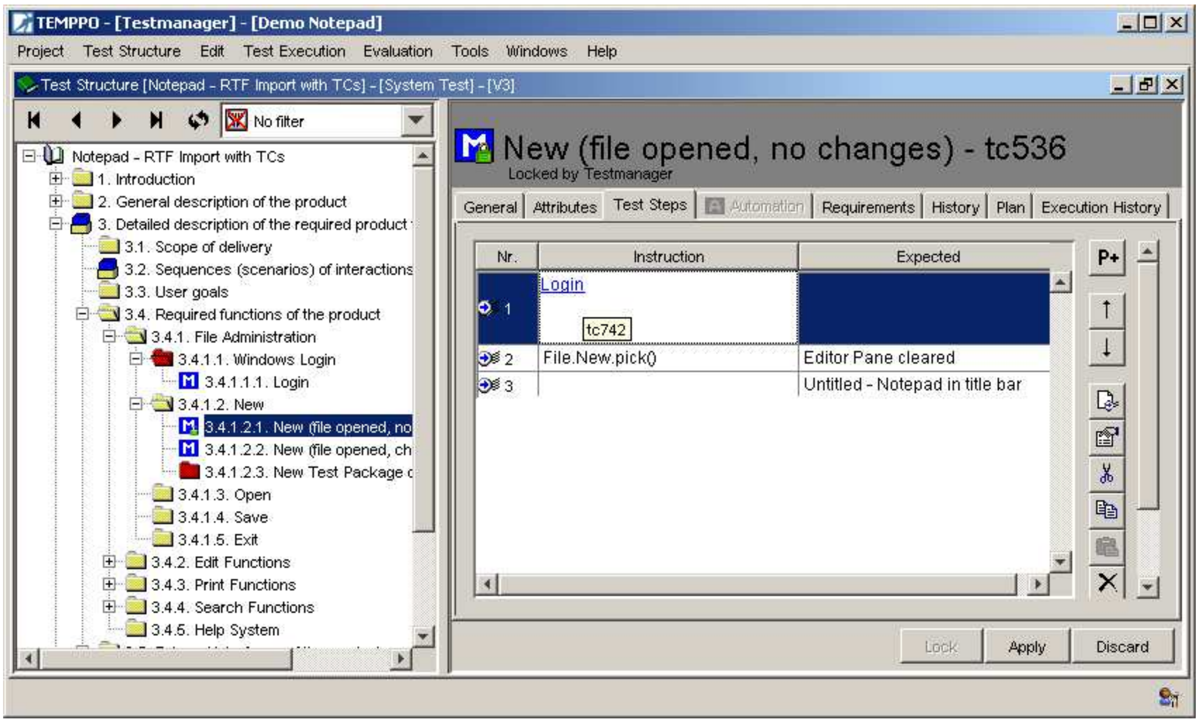


Figure 208 - Select test case

After pressing the **OK** button, the selected test case is displayed by its **ID**, when moving the mouse over the hyperlink.

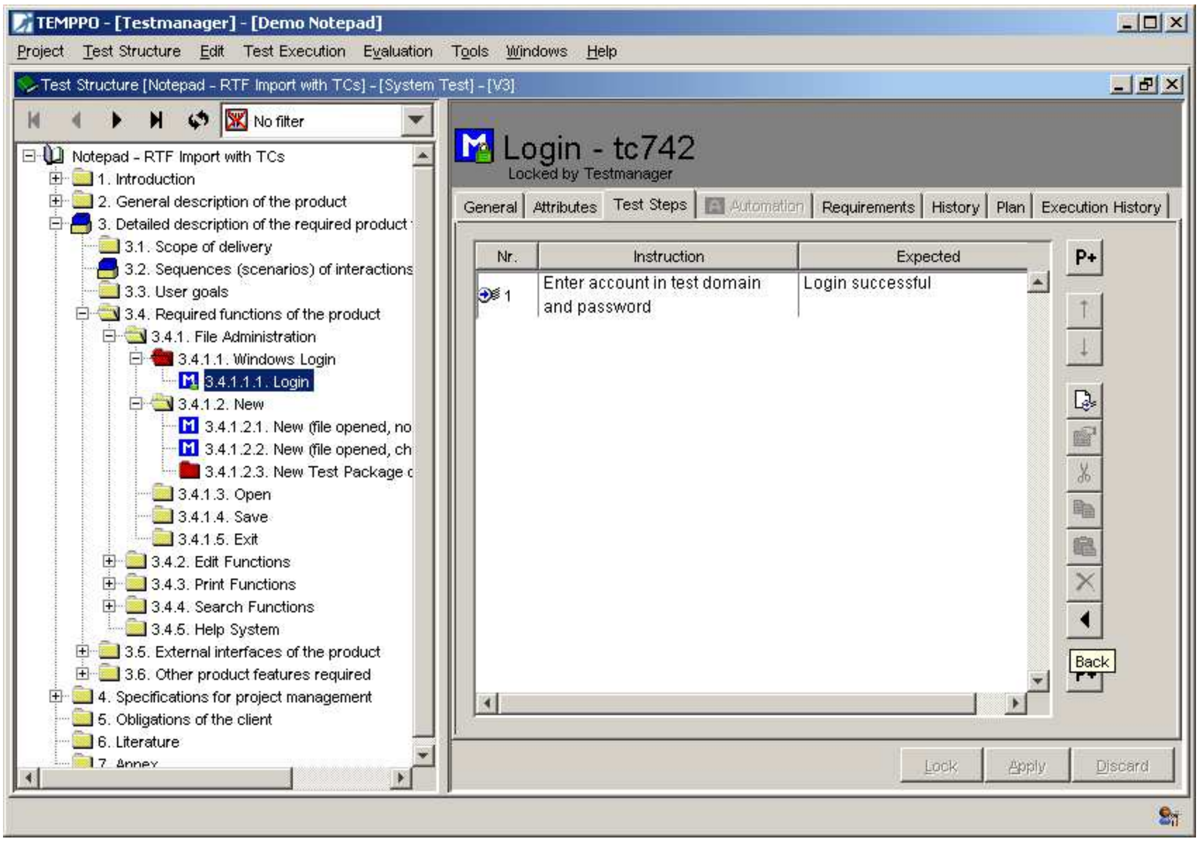

TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required product
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. Windows Login
3.4.1.1.1. Login
3.4.1.2. New
3.4.1.2.1. New (file opened, no
3.4.1.2.2. New (file opened, ch
3.4.1.2.3. New Test Package c
3.4.1.3. Open
3.4.1.4. Save
3.4.1.5. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.5. Help System
General
Attributes
Test Steps
Automation
Requirements
History
Plan
Execution History
P+
Lock
Apply
Discard

New (file opened, no changes) - tc536
Locked by Testmanager

Nr.	Instruction	Expected
1	Login tc742	
2	File.New.pick()	Editor Pane cleared
3		Untitled - Notepad in title bar

Figure 209 - Referenced Test case ID

If you click on the hyperlink, the “destination” test cases is selected automatically. Additionally a **Back** button (see Figure 210) is shown to return to the “source” test case.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required product
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. Windows Login
3.4.1.1.1. Login
3.4.1.2. New
3.4.1.2.1. New (file opened, no
3.4.1.2.2. New (file opened, ch
3.4.1.2.3. New Test Package c
3.4.1.3. Open
3.4.1.4. Save
3.4.1.5. Exit
3.4.2. Edit Functions
3.4.3. Print Functions
3.4.4. Search Functions
3.4.5. Help System
3.5. External interfaces of the product
3.6. Other product features required
4. Specifications for project management
5. Obligations of the client
6. Literature
7. Annex
Login - tc742
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
P+
Back
Lock Apply Discard

Nr.	Instruction	Expected
1	Enter account in test domain and password	Login successful

Figure 210 –Referenced test case with “Back” button

3.3.2.2 Edit hyperlinks

If you select a hyperlink (web, file system, upload) and activate the context sensitive menu (mouse over hyperlink) and choose the menu hyperlink, Figure 200 is shown, where you can change the

- link into the web
- link into the file system
- upload
- link for the auto tool
- link with a command
- link to another test case

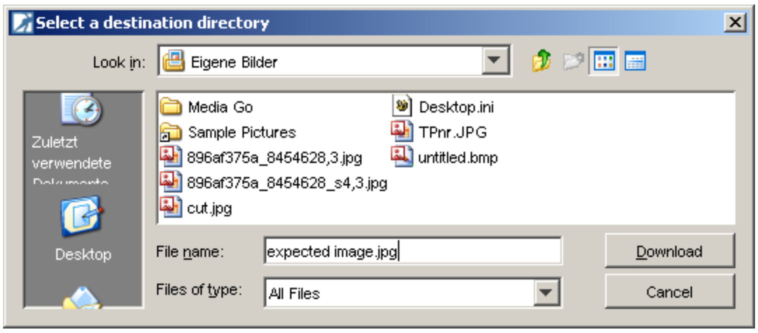
3.3.2.3 Remove hyperlinks

If you select a hyperlink (web, file system, upload) and activate the context sensitive menu (mouse over hyperlink) and choose the menu **Remove hyperlink**, the hyperlink is removed. But take care that an upload remains in the database.

ⓘ **To remove obsolete uploads, see chapter 5.4.5**

3.3.2.4 Download an upload

If you select a hyperlink (upload) and activate the context sensitive menu (mouse over hyperlink) and choose the menu **Download**, Figure 211 is shown to select a directory for the file download.



Look in: Eigene Bilder

File name: expected image.jpg

Files of type: All Files

Figure 211 - Destination selection

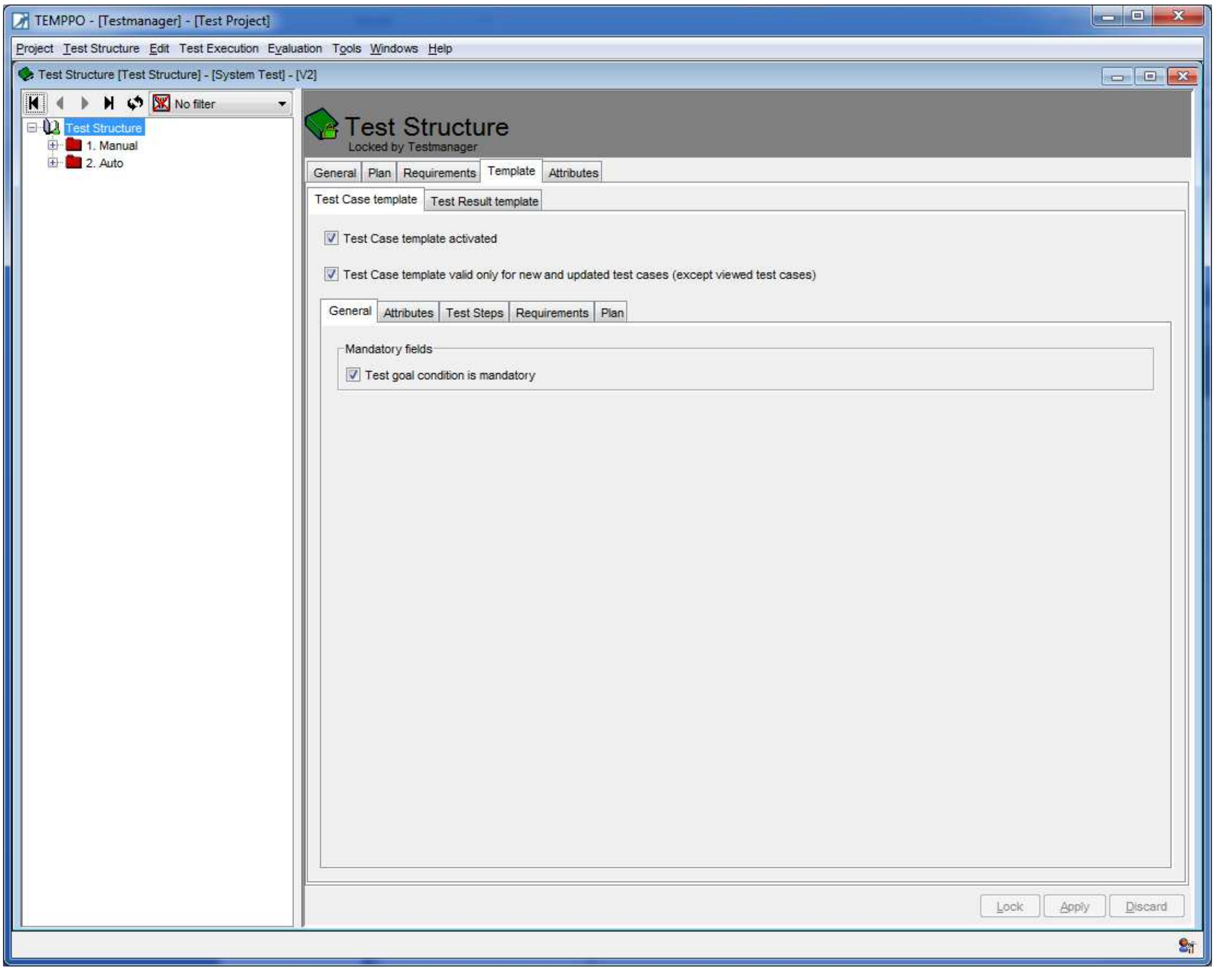
3.3.3 Test Case Templates

Templates can be defined for test structures and test suites. For each test structure exactly 1 test case template can be defined. For test suites belonging to a test structure a template can be configured as well.

In the tab **"Test Case Template"** the test manager defines a basic test case. He can configure **standard attributes**, define **mandatory attributes** or determines if a test case has to be at least **one link to a requirement**. If these conditions are not fulfilled the test case cannot be saved. For details see chapter 3.3.3.1

In the tab **"Test Result Template"** the test manager configures result attributes for test case. If a tester records a result and a test result template is activated, he has to add values for the **mandatory attributes**. For details see chapter 3.3.3.2.

3.3.3.1 Test case template



☑ Test Case template activated

☑ Test Case template valid only for new and updated test cases (except viewed test cases)

☑ Test goal condition is mandatory

Figure 212 – Test Case Template: General

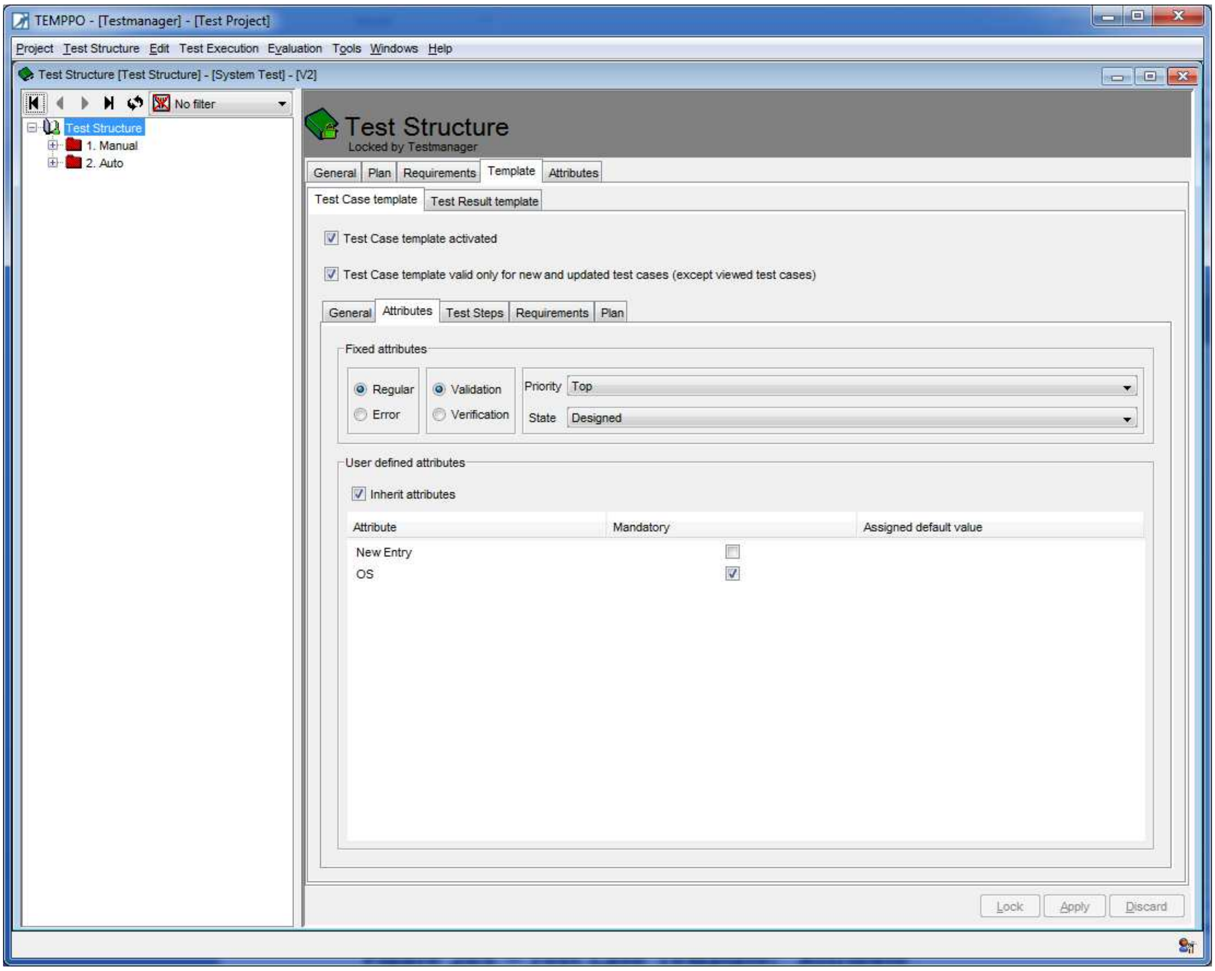
A Test Case template (TCT) supports testers to create and update test cases in complete manner. For a TCT you can define

- **Mandatory fields** (user has to enter a **test goal, precondition, postcondition,** see Figure 212 and Figure 214
- **Default values** for the **fixed attributes**, e.g. priority has to be "In work", if a test case is created, see Figure 213
- **Inheritance of UDAs** and/or **defining UDAs as mandatory** or with a default value, see Figure 213
- **Inheritance of requirements** and/or **linking at least 1 mandatory requirement**, see Figure 215
- Default values for a **planned tester** and a **planned execution time**, see Figure 216

A TCT is concerned in two cases in TEMPPO: When creating a new test case and locking/editing an existing one.

You can activate a TCT for the use case of creating and updating test cases but excluding the use case of viewing: clicking trough (= locking – unlocking) test cases.

You can activate a TCT (check) also for viewing existing test cases. E.g., if you activate the TCT after creating some test cases that do not fulfill the conditions and view them later, you cannot click another test case without fulfilling the TCT (=leave a locked TC).


TEMPPO - [Testmanager] - [Test Project]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Test Structure] - [System Test] - [V2]
No filter
Test Structure
1. Manual
2. Auto
Test Structure
Locked by Testmanager
General Plan Requirements Template Attributes
Test Case template Test Result template
General Attributes Test Steps Requirements Plan
Lock Apply Discard

☑ Test Case template activated

☑ Test Case template valid only for new and updated test cases (except viewed test cases)

Fixed attributes

● Regular ○ Error

● Validation ○ Verification

Priority: Top

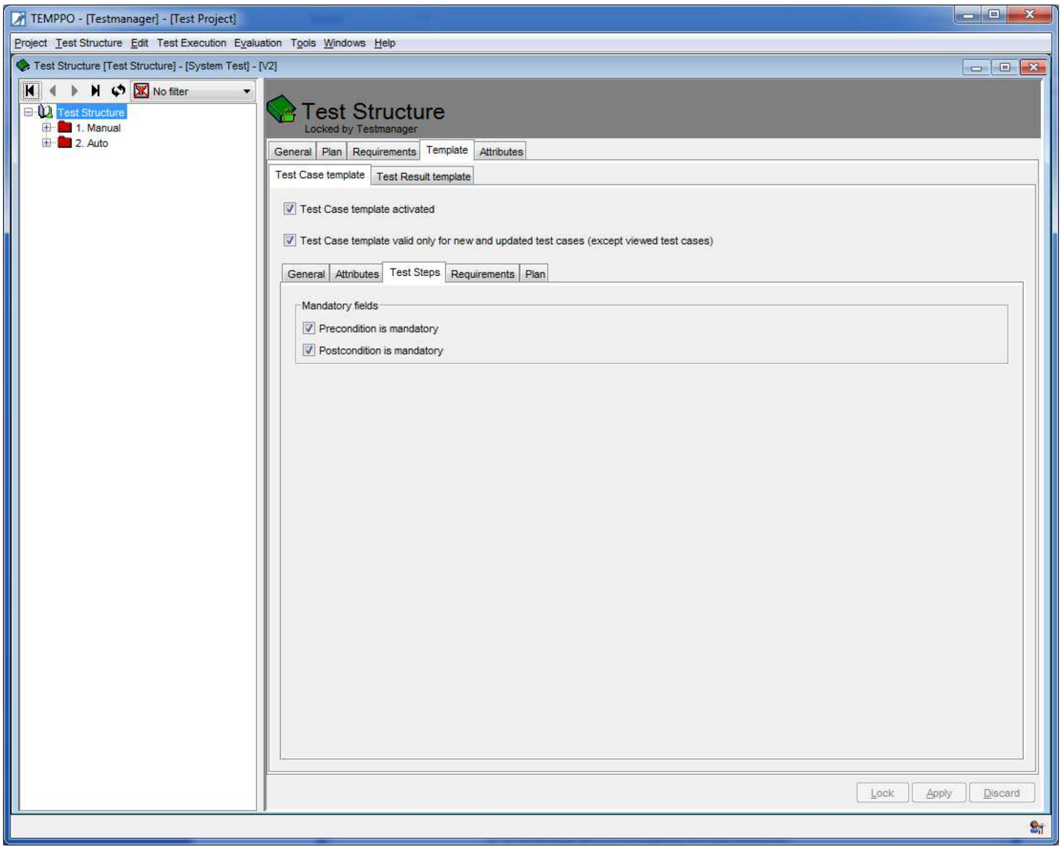
State: Designed

User defined attributes

☑ Inherit attributes

Attribute	Mandatory	Assigned default value
New Entry	□	
OS	☑	

Figure 213 – Test Case Template: Attribute


TEMPPO - [Testmanager] - [Test Project]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Test Structure] - [System Test] - [V2]
No filter
Test Structure
1. Manual
2. Auto
Test Structure
Locked by Testmanager
General Plan Requirements Template Attributes
Test Case template Test Result template
General Attributes Test Steps Requirements Plan
Lock Apply Discard

☑ Test Case template activated

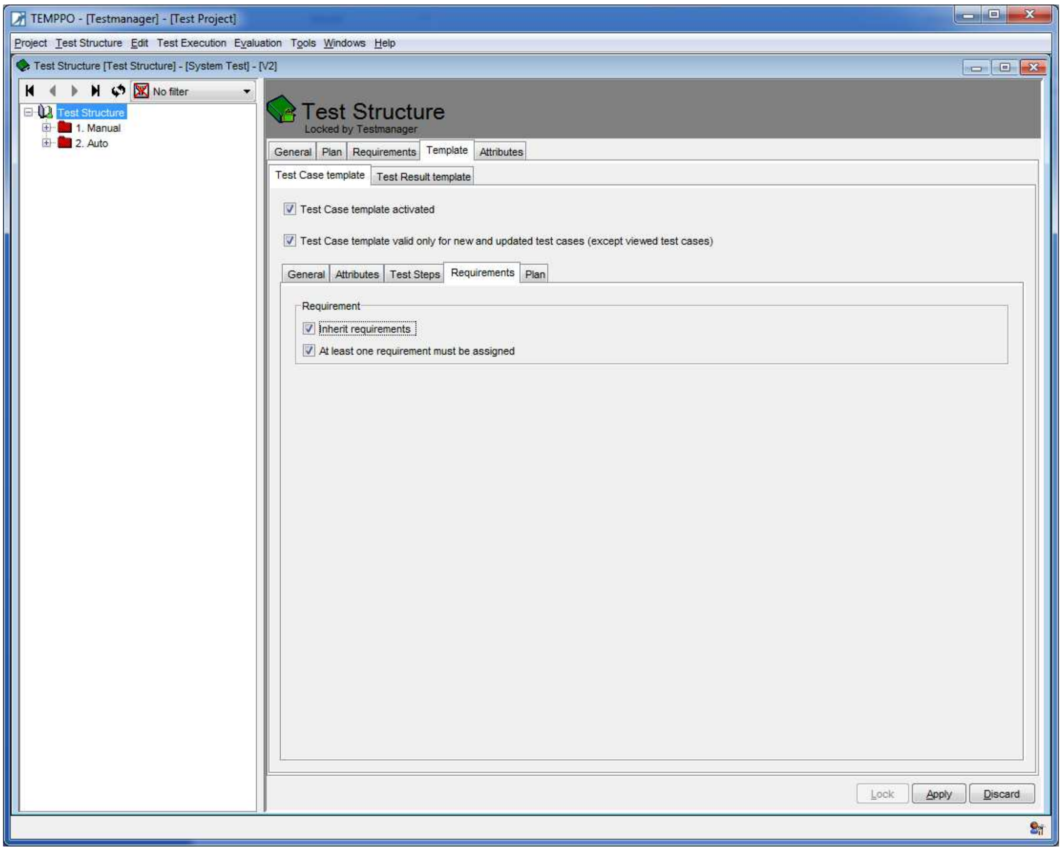
☑ Test Case template valid only for new and updated test cases (except viewed test cases)

Mandatory fields

☑ Precondition is mandatory

☑ Postcondition is mandatory

Figure 214 – Test Case Template: Test Step


TEMPPO - [Testmanager] - [Test Project]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Test Structure] - [System Test] - [V2]
No filter
Test Structure
1. Manual
2. Auto
Test Structure
Locked by Testmanager
General Plan Requirements Template Attributes
Test Case template Test Result template
General Attributes Test Steps Requirements Plan
Lock Apply Discard

☑ Test Case template activated

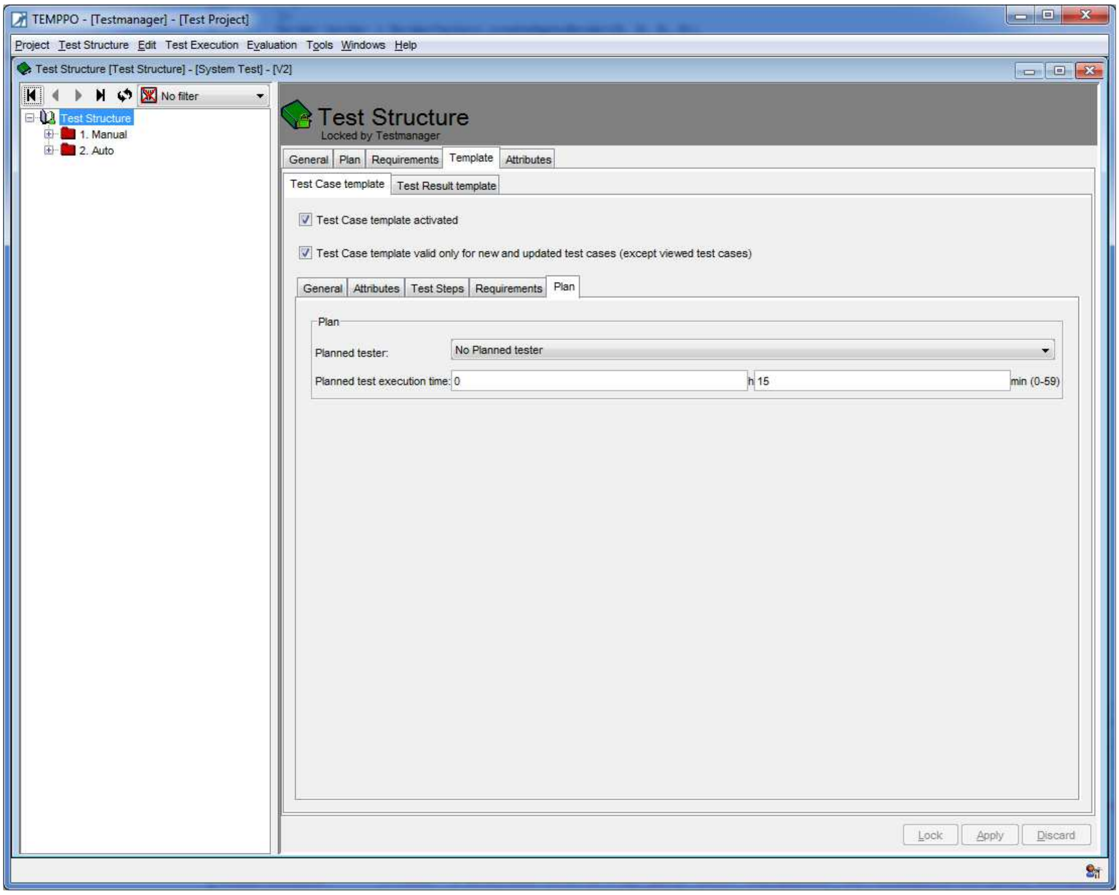
☑ Test Case template valid only for new and updated test cases (except viewed test cases)

Requirement

☑ Inherit requirements

☑ At least one requirement must be assigned

Figure 215 – Test Case Template: Requirement


TEMPPO - [Testmanager] - [Test Project]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Test Structure] - [System Test] - [V2]
No filter
Test Structure
1. Manual
2. Auto
Test Structure
Locked by Testmanager
General Plan Requirements Template Attributes
Test Case template Test Result template
General Attributes Test Steps Requirements Plan
Lock Apply Discard

☑ Test Case template activated

☑ Test Case template valid only for new and updated test cases (except viewed test cases)

Plan

Planned tester: No Planned tester

Planned test execution time: 0 h 15 min (0-59)

Figure 216 – Test Case Template: Plan

3.3.3.1.1 Create new test case

If a TCT is activated and a new test case is created, the values of TCT are applied:

- fixed attributes are set automatically.
- UDAs with default values (but without ticked checkbox **Mandatory**) are assigned automatically
- inheritance flag of requirements is (de)activated
- inheritance flag of UDAs is (de)activated
- default values for planned tester and planned execution time are set automatically

The new created test case can only be saved (button **Apply**, clicking another test case),

- if a precondition is entered and the corresponding checkbox in TCT (see Figure 214 – Test Case Template: Test Step) is ticked
- if a postcondition is entered and the corresponding checkbox in TCT (see Figure 214 – Test Case Template: Test Step) is ticked
- if a test goal is entered and the corresponding checkbox in TCT (see Figure 212) is ticked
- if at least one requirement is linked and the corresponding checkbox in TCT (see Figure 215) is ticked
- if an UDA plus value is assigned and the corresponding checkbox in TCT (see Figure 213) is ticked.

3.3.3.1.2Unlock/save existing test case

An existing test case can be unlocked/saved without message,

- if a precondition is entered and the corresponding checkbox in TCT (see Figure 214) is ticked.
- if a postcondition is entered and the corresponding checkbox in TCT (see Figure 214) is ticked.
- if a test goal is entered and the corresponding checkbox in TCT (see Figure 212) is ticked.
- if checkbox **Inherit** in tab **Requirement** is activated and the corresponding checkbox in TCT (see Figure 215) is ticked.
- if at least one requirement is linked and the corresponding checkbox in TCT (see Figure 215) is ticked.
- if checkbox **Inherit** in tab **Attributes** is activated and the corresponding checkbox in TCT (see Figure 213) is ticked.
- If an UDA plus value is assigned and the corresponding checkbox in TCT (see Figure 213) is ticked.

ⓘ **If a TCT is activated and a test case with missing e.g. default attribute values is edited, those are not added automatically!**

3.3.3.1.3Test case template and filter

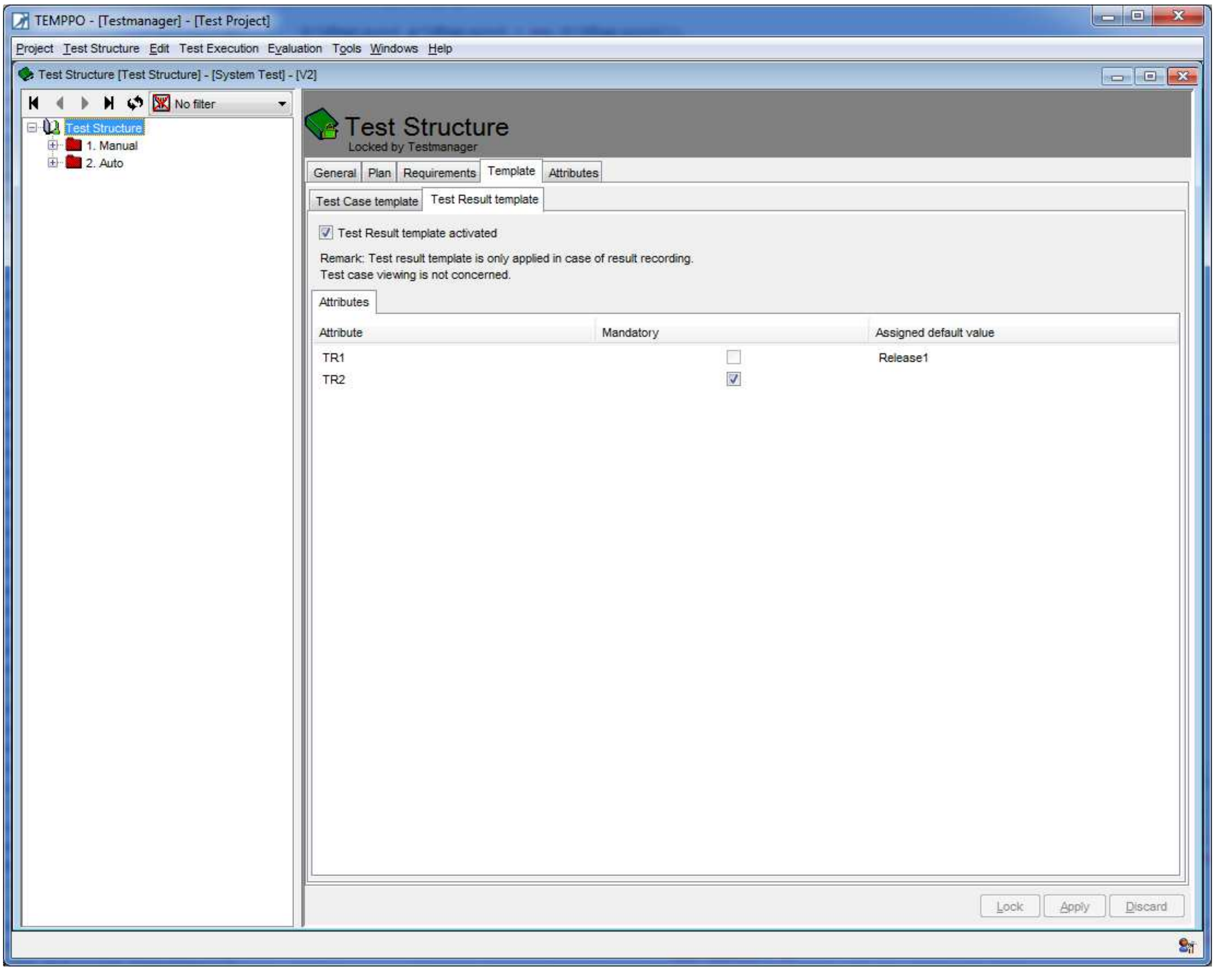
If

- a TCT is activated and
- a filter is set and
- a new TC should be created and
- the test case don’t fulfill the filter criterion

the test case cannot be created. You have to set the filter to “No filter” and start again the creation progress.

3.3.3.2 Test result templates

A test result template supports you in test case execution. You can define a test result as mandatory or set a default value, if a result is recorded.



☑ Test Result template activated

Remark: Test result template is only applied in case of result recording.
Test case viewing is not concerned.

Attribute	Mandatory	Assigned default value
TR1	□	Release1
TR2	☑	

Figure 217 – Test result template

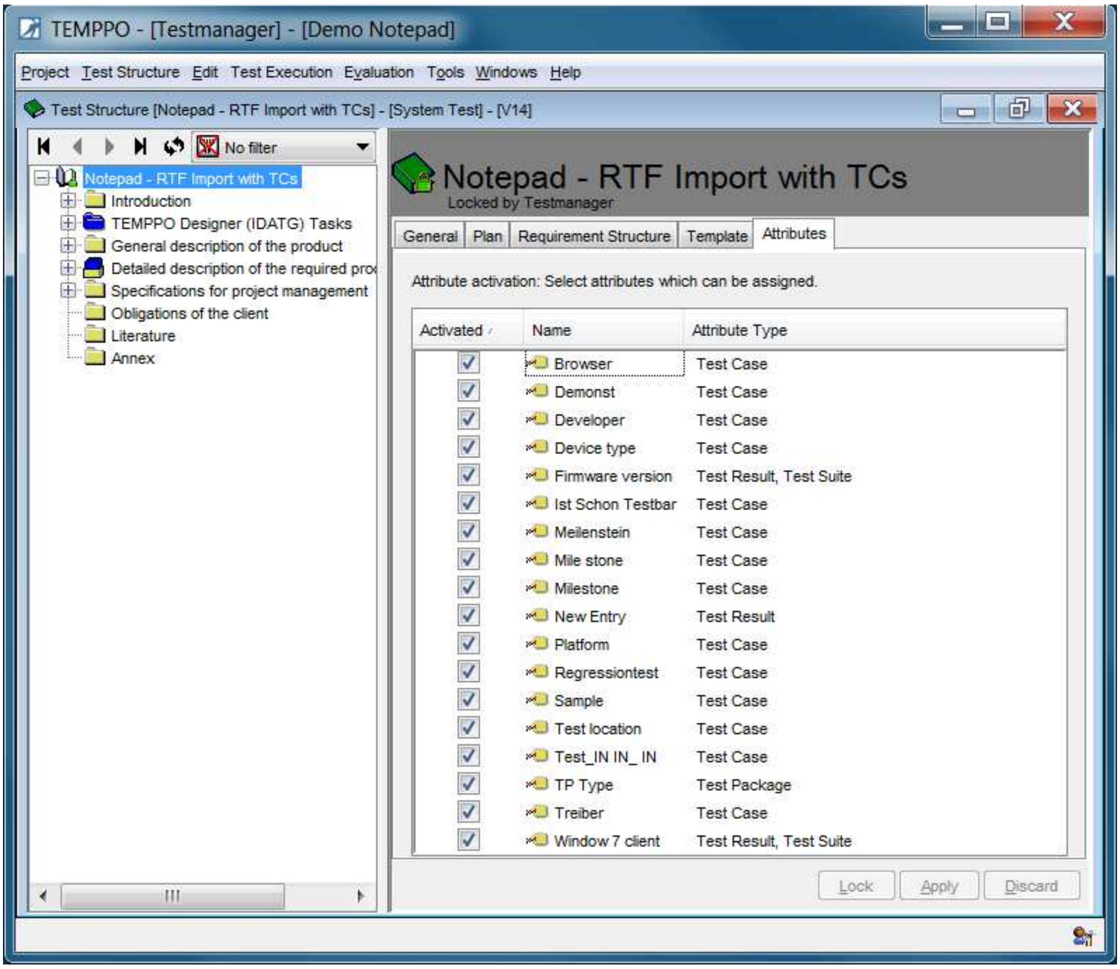
In Figure 217, test result attributes are defined. If you record a test result “TR1 = Release1” is assigned automatically and you have to specify a value for TR2. Otherwise the test case cannot be saved.

3.3.4 Attribute activation

In some cases, attributes shouldn’t be assigned to test cases anymore because they are invalid in future. In TEMPPO, available attributes can be restricted for assignments, but assignments from past are still visible and can be used e.g. for filtering.

Attribute activation can be defined for each test structure in tab **Attributes**, see also Figure 218. If the check box for an attribute is greyed, the attribute is set in template.

Only Superusers and TEMPPO Key Users are allowed to deactivate attributes.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V14]
No filter
Notepad - RTF Import with TCs
Introduction
TEMPPO Designer (IDATG) Tasks
General description of the product
Detailed description of the required pro
Specifications for project management
Obligations of the client
Literature
Annex
Notepad - RTF Import with TCs
Locked by Testmanager
General
Plan
Requirement Structure
Template
Attributes
Lock
Apply
Discard

Attribute activation: Select attributes which can be assigned.

Activated	Name	Attribute Type
☑	Browser	Test Case
☑	Demonst	Test Case
☑	Developer	Test Case
☑	Device type	Test Case
☑	Firmware version	Test Result, Test Suite
☑	Ist Schon Testbar	Test Case
☑	Meilenstein	Test Case
☑	Mile stone	Test Case
☑	Milestone	Test Case
☑	New Entry	Test Result
☑	Platform	Test Case
☑	Regressiontest	Test Case
☑	Sample	Test Case
☑	Test location	Test Case
☑	Test_IN IN_ IN	Test Case
☑	TP Type	Test Package
☑	Treiber	Test Case
☑	Window 7 client	Test Result, Test Suite

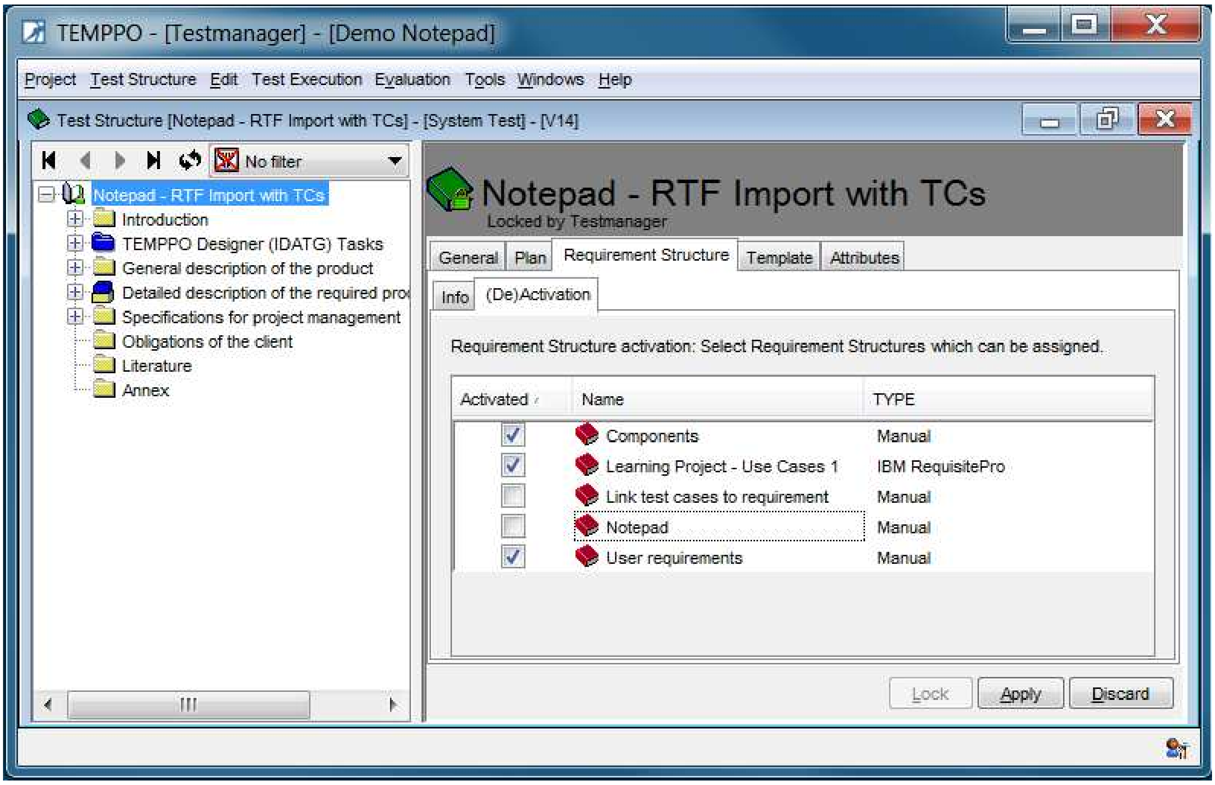
Figure 218 – Attribute activation

3.3.5 Requirement (Structure) activation

In some cases, requirements shouldn't be assigned to test cases anymore because they are invalid in future. In TEMPPO, available requirement structures can be restricted for assignments, but assignments from past are still visible and can be used e.g. for filtering.

Requirement Structure activation can be defined for each test structure in tab **Requirement Structure -> (De)Activation**, see also Figure 218. If the check box for a requirement structure is greyed, the attribute is set in template.

Only Superusers and TEMPPO Key Users are allowed to deactivate requirement structures.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V14]
No filter
Notepad - RTF Import with TCs
Introduction
TEMPPO Designer (IDATG) Tasks
General description of the product
Detailed description of the required pro
Specifications for project management
Obligations of the client
Literature
Annex
Notepad - RTF Import with TCs
Locked by Testmanager
General
Plan
Requirement Structure
Template
Attributes
Info
(De)Activation
Lock
Apply
Discard

Requirement Structure activation: Select Requirement Structures which can be assigned.

Activated	Name	TYPE
☑	Components	Manual
☑	Learning Project - Use Cases 1	IBM RequisitePro
☐	Link test cases to requirement	Manual
☐	Notepad	Manual
☑	User requirements	Manual

Figure 219 - Requirement Structure Activation

3.3.6 Move Ownership

Owner of an item is the user who creates this item. Move Ownership is offered for the following items:

- **Requirement**
- **Test Level**
- **Test Category**
- **Attribute**
- **Upload**
- **User Fields**
- **Test Case**
- **Test Package**
- **Test Structure**
- **Test Suite**

For Move Ownership of Projects see 5.5.6.

For Metadata:

Select the corresponding metadata in the TEMPPO Administrator application. Select the metadata you want to change and press the button **Move ownership**.

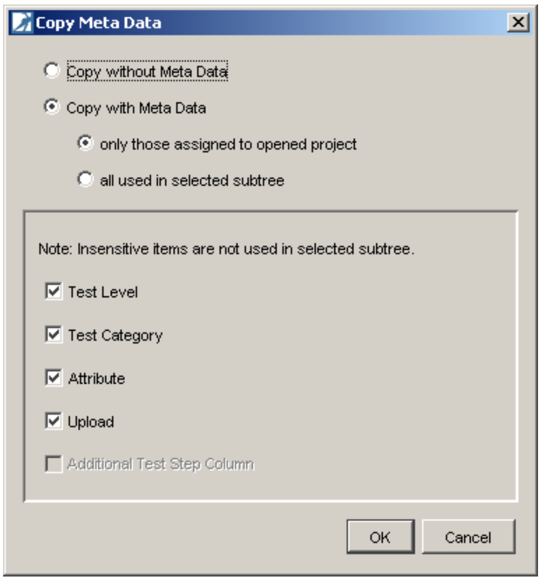
For Test Case, Test Package, Test Structure and Test Suite:

Select the item and click the menu item

A list of all TEMPPO users is presented to the current user (see Figure 320 on page 248). A new owner of the selected item can be chosen (see also 5.5.6).

3.3.7 Map of Meta Data (project wide copy)

If parts of a test structure are copied from one project A to another project B, the related metadata can be copied, too. At the beginning of the paste action the user is asked, if metadata should be mapped, too (see Figure 220).



○ Copy without Meta Data

● Copy with Meta Data

● only those assigned to opened project

○ all used in selected subtree

Note: Insensitive items are not used in selected subtree.

☑ Test Level

☑ Test Category

☑ Attribute

☑ Upload

□ Additional Test Step Column

Figure 220 – Copy Meta Data

There you can choose from three possibilities to paste:

- Copy without Meta Data: Only the selected part of the structure is copied, but not the metadata.
- Copy with Meta Data:
    - Only those assigned to opened project: The selected part are copied only with those metadata which are already assigned to the target project.
    - All used in selected sub tree: The selected part with all assigned metadata is pasted to the project B. Metadata which is not yet assigned to the target project are implicitly assigned to it.

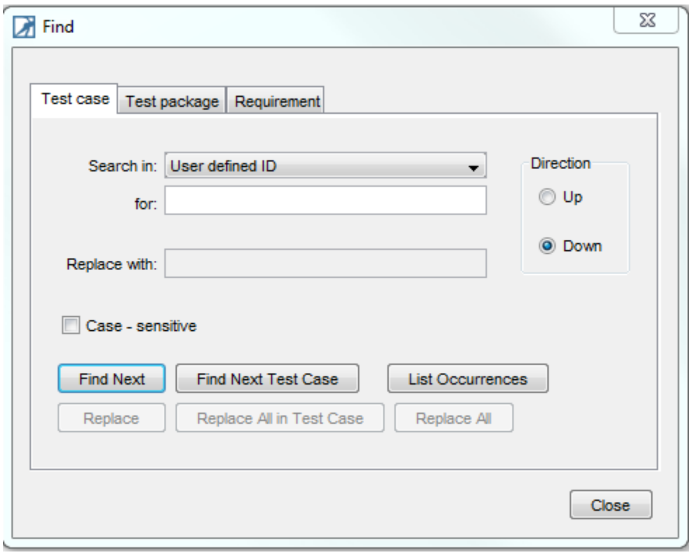
The third possibility is disabled, if the user has not the right "Copy All".

After opening the window, the metadata types (checkboxes) are selected which are used in the copied sub tree. You can also un-/ select each metadata which should be pasted.

By clicking **OK** the items are added to the test structure.

3.3.8 Find and Replace

TEMPPO offers a find and replace mechanism. By pressing **F3** or activating the menu item **Edit > Find and Replace** Figure 221 is shown.



Search in: User defined ID

for:

Replace with:

Direction: ○ Up ● Down

□ Case - sensitive

Figure 221 – Find and Replace

The find/replace dialog includes two tabs, whereas the first one is for searching in **test cases**, **test packages and for requirements**.

Both tabs provide the same features, the only different between this two tabs is the different search-content.

A **test case** in a test structure can be found via searching for

- **user defined ID**
- **name**
- **test goal**
- **precondition**
- **postcondition**
- **test step columns (instruction, input, expected)**
- **user defined fields.**

A test case in a test suite can be found additionally by its **tester's comment**, **actual result** and **bug ID**. Only the text fields tester's comment, actual result and bug ID can be replaced in test suites.

A test package in a test structure can be found via searching for

- **name**
- **description**
- **precondition**
- **postcondition.**

The search criteria can be specified using the wildcards * (0..n arbitrary characters) or/and ? (exactly one arbitrary character) and is case sensitive.

The replace function searches for the test case and then the searched part will be replaced to the entered name.

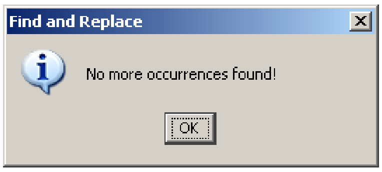
Example: You want to change the name of the test cases. The part "Test Case" should be changed to the German name "Testfall", so you select "Name" in **Search in** and write in the text field **for** "Test Case". In the **Replace with** – text field you write "Testfall".

A searching direction can be chosen, too. The searching algorithm starts from the selected node in the tree. Is the tree root selected, the whole tree is searched whatever direction is selected.

3.3.8.1.1 Find (Test Case / Test Package)

There are three possibilities for finding test case(s)/ package(s):

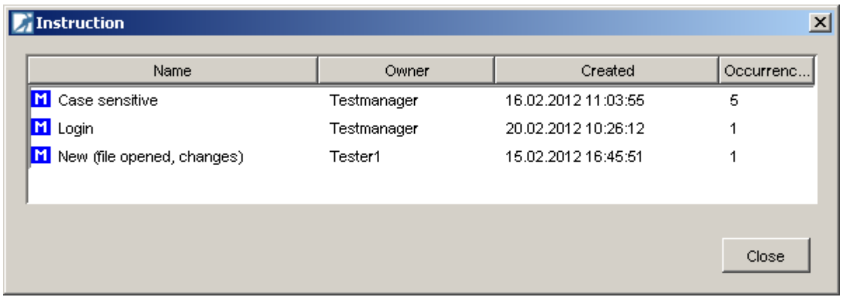
- **Find Next:** With clicking Find Next the first test case/package is searched and selected in the tree. The first occurrence is marked, too. With the next click, the next occurrence is marked. If there are no more occurrences in the test case, the next test case/package would be searched. If there is no element (anymore) a message is shown (see Figure 222).
- **Find Next Test Case/package:** With clicking Find Next Test Case/Package the first test case/package is searched selected in the tree. The first occurrence is marked, too. With the next click, the next test case/package is searched. If there is no element (anymore) a message is shown (see Figure 222).



No more occurrences found!

Figure 222 – No more occurrences found

- **List occurrences**: With the button **List occurrences** the searching result is shown in a list (see Figure 223). When clicking a test case/package in the list, this requirement is selected in the tree.



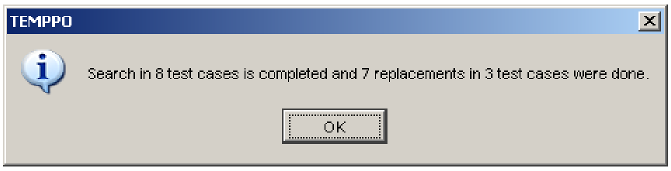
Name	Owner	Created	Occurrenc...
Case sensitive	Testmanager	16.02.2012 11:03:55	5
Login	Testmanager	20.02.2012 10:26:12	1
New (file opened, changes)	Tester1	15.02.2012 16:45:51	1

Figure 223 – List occurrences

3.3.8.1.2 Replace (Test Case / Test Package)

There are also two possibilities for replacing:

- **Replace**: At first a test case/package is searched and selected. The user has to click again for replacing the occurrence and then the next one is searched. If the test case/package is locked, the test case/package isn't changed, e.g. the next one is searched.
- **Replace All in Test Case/package**: To use this feature a test case/package have to be selected first otherwise the button is disabled. At first a test case is searched and selected. The user has to click again for replacing all occurrences in this test case/package. If the test case/package is locked, the test case/package isn't changed.
- **Replace All**: The selected field of all test cases/packages is changed which contains the text. After replacing all, a message is shown (see Figure 224) with the number of replaced test case and number of test case/package which couldn't be replaced, because of locks, etc.



TEMPPO

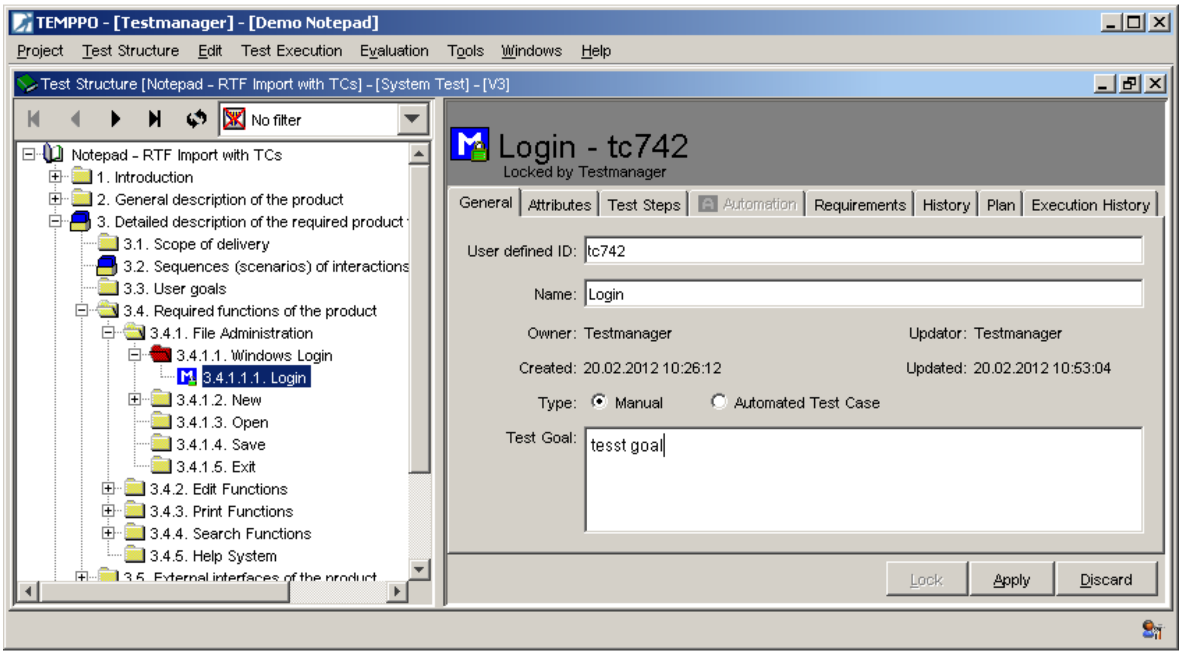
Search in 8 test cases is completed and 7 replacements in 3 test cases were done.

Figure 224 – Replace result

ⓘ **The replace – functionality is disabled, if the user has not the right for changing test case(s) in test structure / test suite.**

3.3.9 Apply / Discard

When working with test structures or test suites, you can save your changes within test cases or test packages using the button Apply. By pressing button Discard, your changes will be undone after a confirmation message. The buttons are only enabled, if the selected node has been changed since it was loaded.



TEMPPO - [Testmanager] - [Demo Notepad]

Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]

Login - tc742
Locked by Testmanager

User defined ID: tc742

Name: Login

Owner: Testmanager

Updator: Testmanager

Created: 20.02.2012 10:26:12

Updated: 20.02.2012 10:53:04

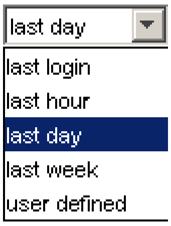
Type: ● Manual ○ Automated Test Case

Test Goal: tesst goal

Figure 225 - Apply / Discard

3.3.10Newsboard

The newsboard informs the user about changes done on the current test structure (version). It can be opened by calling the menu item **Test Structure -> Newsboard**. In **Settings** it can be configured that is it shown automatically after opening a test structure. Since time is going on and newsboard doesn't poll, user has to press the **Refresh** button to get the latest "news".



The display of newsboard can be configured by setting **last login, last hour, last day last week** or **user defined**. If **user defined** is selected, a date with calendar and text fields for time (hh, mm) and the **Apply** button are editable.

ⓘ **If "last login" is selected, the time stamp of user's last logout is chosen!**

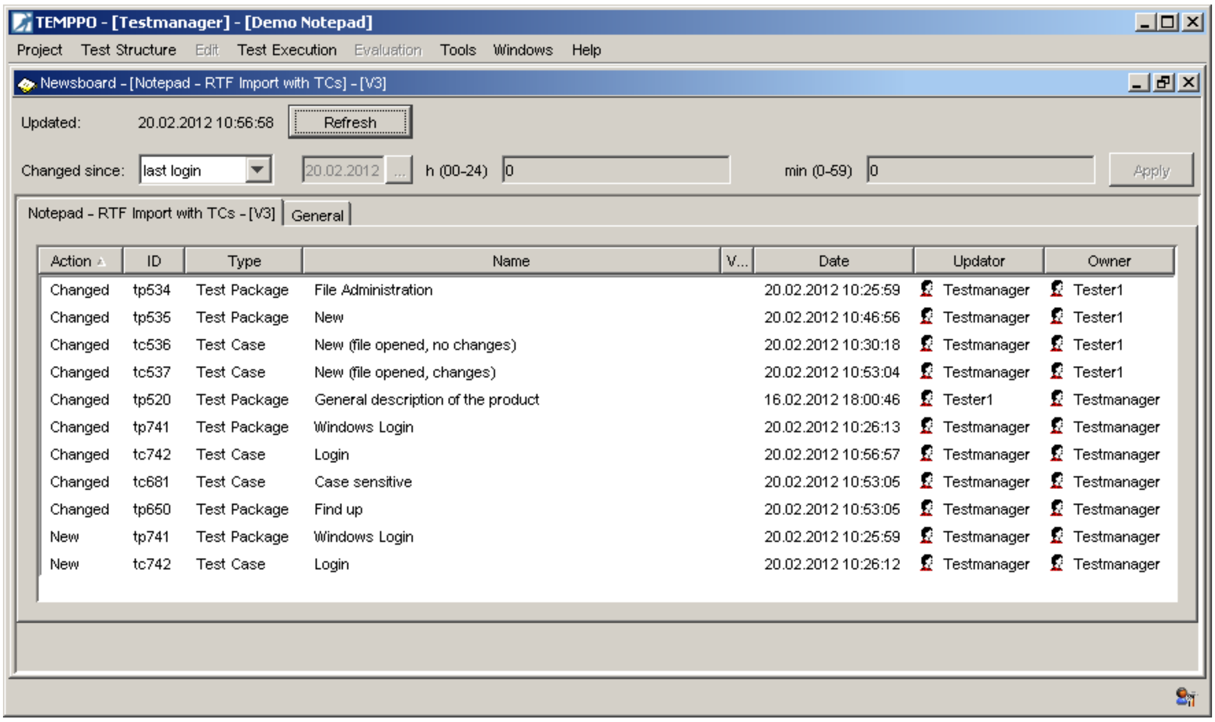
If a newsboard entry is activated, the corresponding requirement is selected automatically, except if one was deleted. In the newsboard table the **action** (new, changed, deleted), **requirement ID**, **requirement name, date of action**, and the **person** are shown.

Newsboard consists of 2 tabs, <**name of test structure plus version**> and **General**. The first one shows all activities (new, change, delete) on test cases and test packages in the current selected test structure version.

On the other hand **General** displays all other information:

New attribute assigned to the project, changed attribute, new attribute value, new test suite, new user, new version.

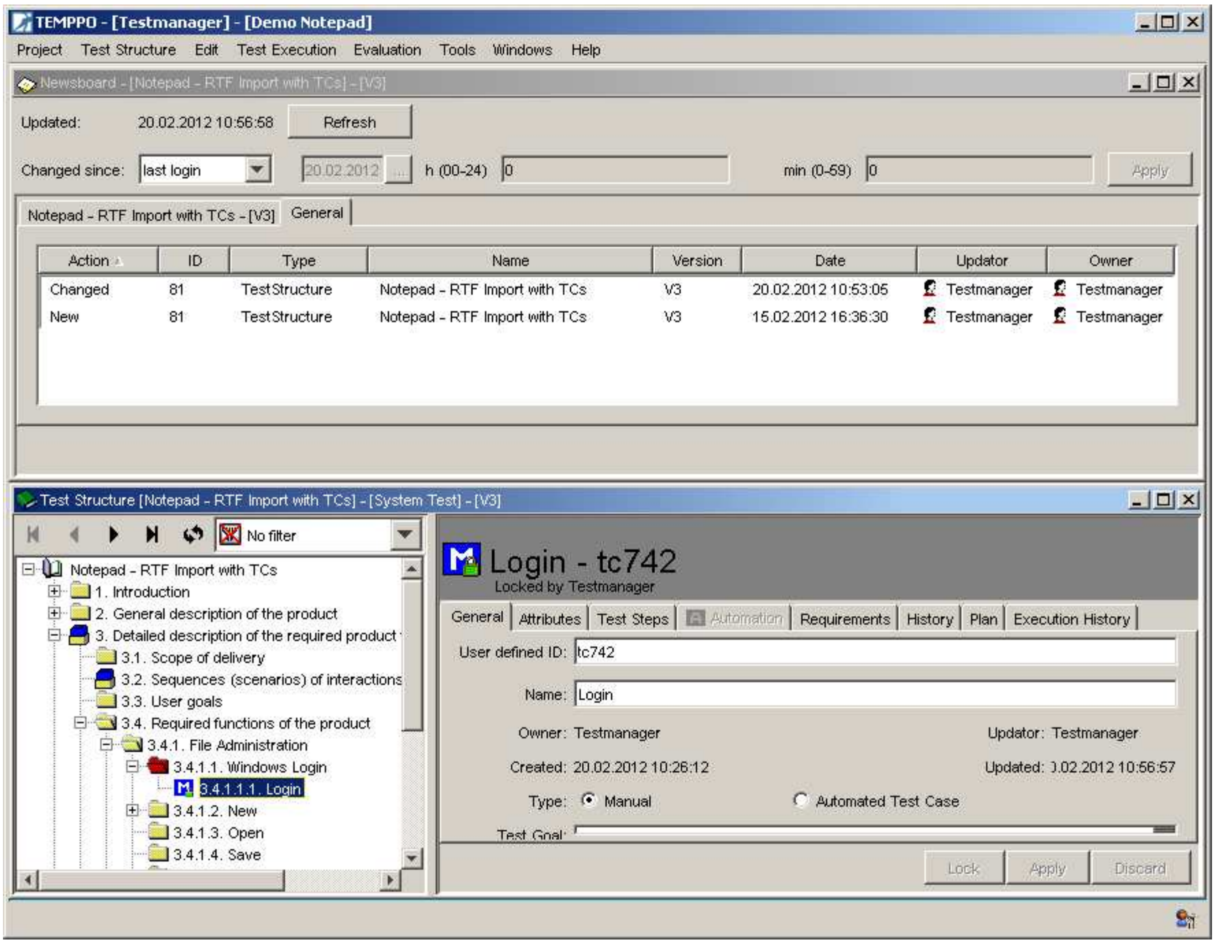
If an item of newsboard is selected, the corresponding one is also highlighted in the test structure (if possible). A deleted item cannot be displayed, of course.


TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Newsboard - [Notepad - RTF Import with TCs] - [V3]
Refresh
Apply
Notepad - RTF Import with TCs - [V3]
General

Updated: 20.02.2012 10:56:58

Changed since: last login 20.02.2012 h (00-24) 0 min (0-59) 0

Action	ID	Type	Name	V...	Date	Updator	Owner
Changed	tp534	Test Package	File Administration		20.02.2012 10:25:59	Testmanager	Tester1
Changed	tp535	Test Package	New		20.02.2012 10:46:56	Testmanager	Tester1
Changed	tc536	Test Case	New (file opened, no changes)		20.02.2012 10:30:18	Testmanager	Tester1
Changed	tc537	Test Case	New (file opened, changes)		20.02.2012 10:53:04	Testmanager	Tester1
Changed	tp520	Test Package	General description of the product		16.02.2012 18:00:46	Tester1	Testmanager
Changed	tp741	Test Package	Windows Login		20.02.2012 10:26:13	Testmanager	Testmanager
Changed	tc742	Test Case	Login		20.02.2012 10:56:57	Testmanager	Testmanager
Changed	tc681	Test Case	Case sensitive		20.02.2012 10:53:05	Testmanager	Testmanager
Changed	tp650	Test Package	Find up		20.02.2012 10:53:05	Testmanager	Testmanager
New	tp741	Test Package	Windows Login		20.02.2012 10:25:59	Testmanager	Testmanager
New	tc742	Test Case	Login		20.02.2012 10:26:12	Testmanager	Testmanager


TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Newsboard - [Notepad - RTF Import with TCs] - [V3]
Refresh
Apply
Notepad - RTF Import with TCs - [V3]
General
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required product
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. Windows Login
3.4.1.1.1. Login
3.4.1.2. New
3.4.1.3. Open
3.4.1.4. Save
Login - tc742
Locked by Testmanager
General
Attributes
Test Steps
Automation
Requirements
History
Plan
Execution History
Test Goal:
Lock
Apply
Discard

Updated: 20.02.2012 10:56:58

Changed since: last login 20.02.2012 h (00-24) 0 min (0-59) 0

Action	ID	Type	Name	Version	Date	Updator	Owner
Changed	81	TestStructure	Notepad - RTF Import with TCs	V3	20.02.2012 10:53:05	Testmanager	Testmanager
New	81	TestStructure	Notepad - RTF Import with TCs	V3	15.02.2012 16:36:30	Testmanager	Testmanager

User defined ID: tc742

Name: Login

Owner: Testmanager Updator: Testmanager

Created: 20.02.2012 10:26:12 Updated:).02.2012 10:56:57

Type: ● Manual ○ Automated Test Case

Figure 226 – Newsboard

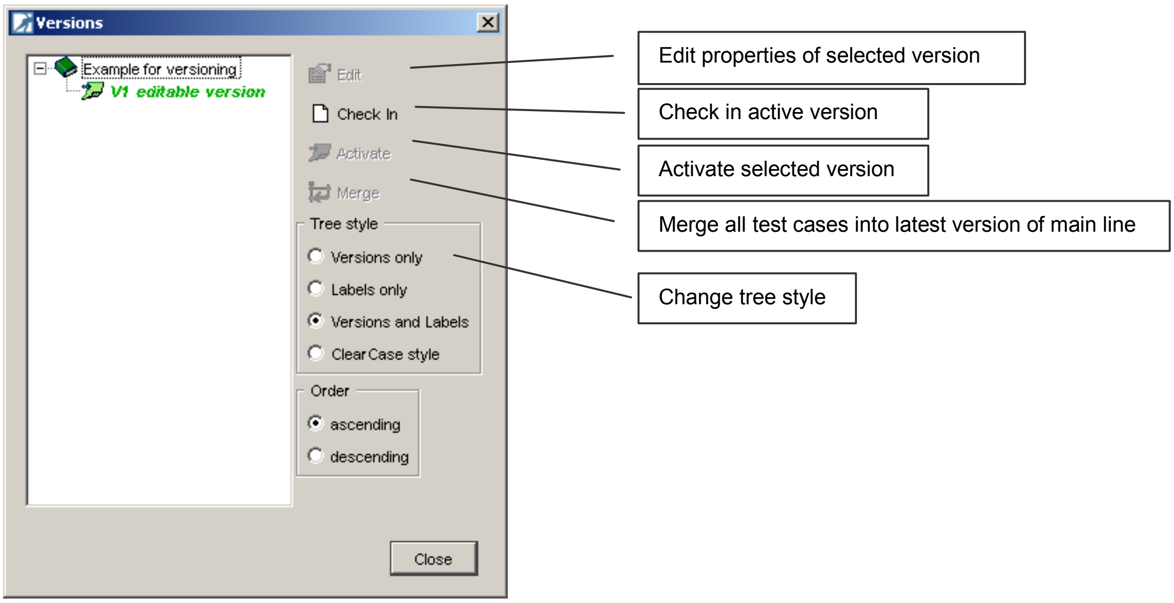
3.3.11Versioning

Configuration management is a task that covers all phases in the test process. TEMPPO does not have a connection to current CM tools like ClearCase or MKS. Supporting a large palette of CM tools from a very powerful but expensive one like ClearCase to a cheap one with restricted functionality would be a challenge. For example ClearCase offers attributes for each administration unit; VSS does not have such commands.

Therefore, the developers of TEMPPO decided to implement an own solution for maintaining versions of test structures with test cases. In TEMPPO it is only possible to check in an entire test structure (=baseline), where “whole” means the hierarchical order of all test packages and test cases as well as their data. In contrast to other CM tools where check in and check out are separate steps, checking in a test structure in TEMPPO will check it out again, i.e. a new editable version is created automatically. Naturally you have the possibility to restore any version of the test structure that has been checked in. Versioning is meaningful when you have to change / extend your test structure due to version changes of your application.

You create and activate versions in TEMPPO using **Test Structure – Version....** The following example should illustrate how the CM features of TEMPPO work.

1. Consider the test structure displayed in Figure 227. Initially there exists just one version of the test structure, version "1" on the main line. In TEMPPO, every last version gets the label "latest", signaling that the test structure in this version is checked out for editing.



Tree style

○ Versions only
○ Labels only
● Versions and Labels
○ ClearCase style

Order

● ascending
○ descending

Edit properties of selected version

Check in active version

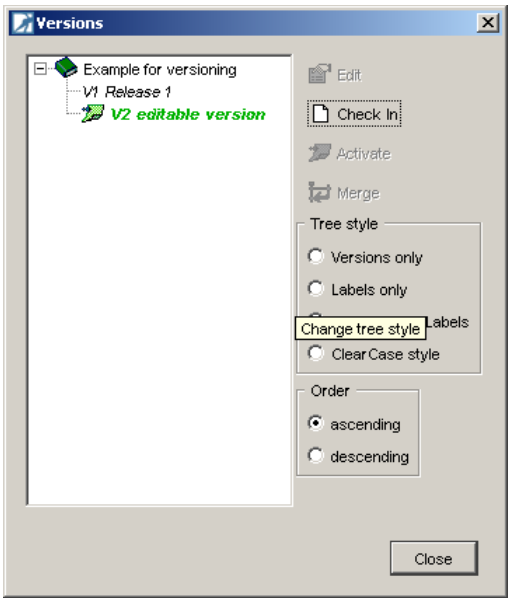
Activate selected version

Merge all test cases into latest version of main line

Change tree style

Figure 227 - CM, Version tree before first check in

If you check in the test structure by using the **Check in active version** button in the Versions dialog you have the possibility to specify a label ("Release 1"). Then the version tree looks like displayed in Figure 228. The test structure gets checked out immediately and version "2" is created. The symbol signals the active version, which is again "latest".



Tree style

○ Versions only
○ Labels only
Labels
○ ClearCase style

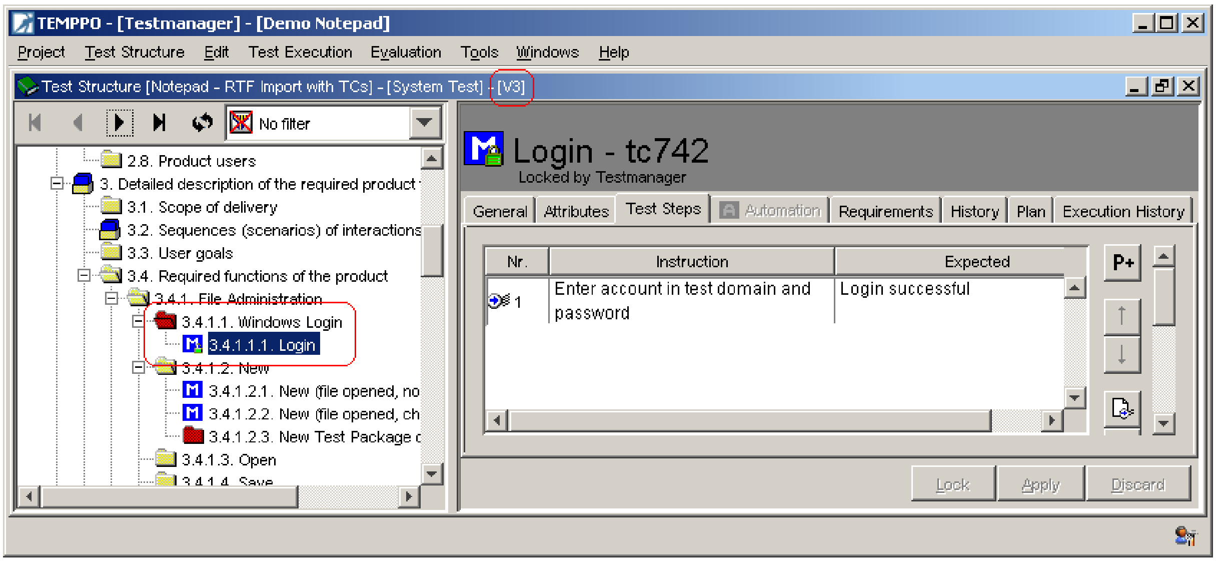
Order

● ascending
○ descending

Figure 228 - CM, Version tree after checking in and labeling

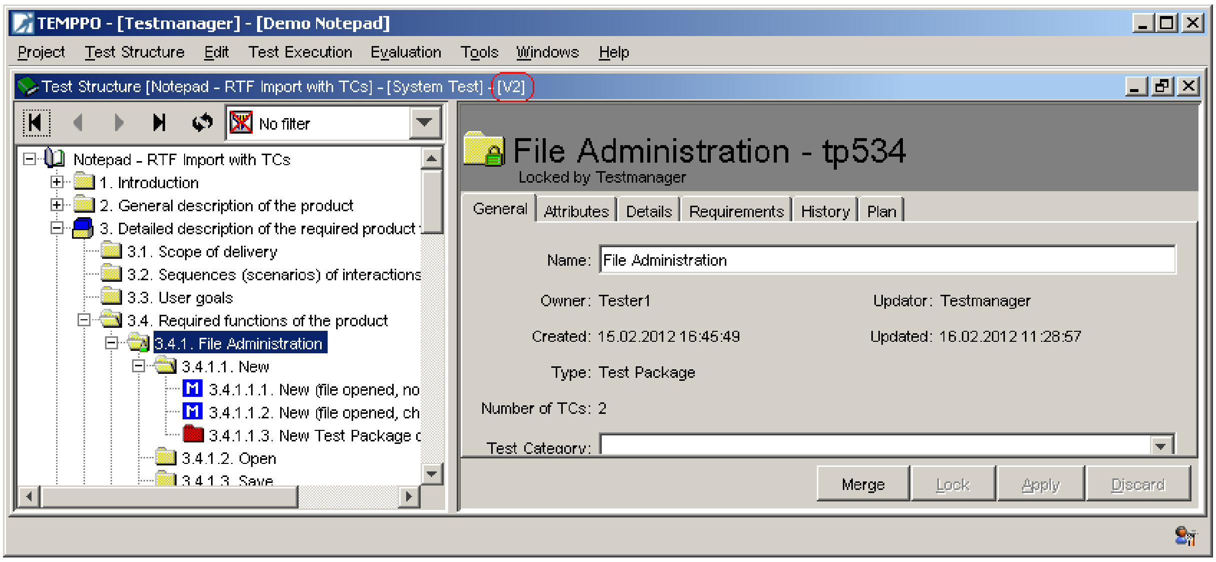
Changes in the latest version of the test structure will now not affect "Release 1". E.g. when inserting a new test package and test case, the new items are only available in the latest version of the test structure (see Figure 229). In the prior versions items are not available, of course (see Figure 230)

ⓘ **You can see the active version of your test structure in the title bar of the test structure explorer.**


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
2.8. Product users
3. Detailed description of the required product
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. Windows Login
3.4.1.1.1. Login
3.4.1.2. New
3.4.1.2.1. New (file opened, no
3.4.1.2.2. New (file opened, ch
3.4.1.2.3. New Test Package c
3.4.1.3. Open
3.4.1.4. Save
Login - tc742
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
P+
Lock Apply Discard

Nr.	Instruction	Expected
1	Enter account in test domain and password	Login successful

Figure 229 – New items in latest structure version


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V2]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required product
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no
3.4.1.1.2. New (file opened, ch
3.4.1.1.3. New Test Package c
3.4.1.2. Open
3.4.1.3. Save
File Administration - tp534
Locked by Testmanager
General Attributes Details Requirements History Plan
Merge Lock Apply Discard

Name: File Administration

Owner: Tester1 Updator: Testmanager

Created: 15.02.2012 16:45:49 Updated: 16.02.2012 11:28:57

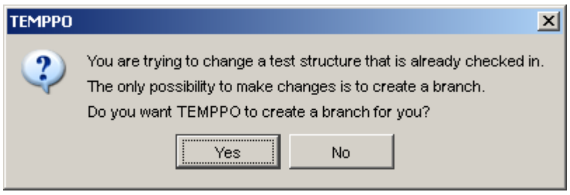
Type: Test Package

Number of TCs: 2

Test Category:

Figure 230 - Items not available in prior versions

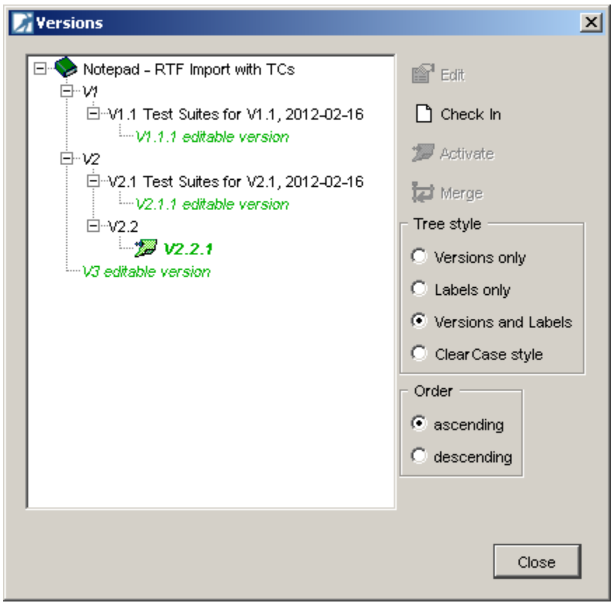
You can restore the version of a test structure by activating it in the versions dialog with button (**Activate selected version**). You cannot change a test structure in a version, which is not "latest". However you can create a branch of a structure version and apply changes to that latest version in the branch. When you activate version 2 (V2) and try to insert a new test package, you will get the following message:



You are trying to change a test structure that is already checked in.
The only possibility to make changes is to create a branch.
Do you want TEMPPO to create a branch for you?

Figure 231 - New branch confirmation

Pressing **No** will abort the action. Pressing **Yes** will create a new V2.2 branch, a branch-latest version of the test structure and the test package in this branch-latest version. The "latest" version in the branch will automatically be activated and can be edited like the latest version in the main line (usually called main-latest). One version can have several branches, which can be created in the same manner.



Versions

- Notepad - RTF Import with TCs
  - V1
    - V1.1 Test Suites for V1.1, 2012-02-16
      - V1.1.1 editable version
  - V2
    - V2.1 Test Suites for V2.1, 2012-02-16
      - V2.1.1 editable version
    - V2.2
      - V2.2.1
  - V3 editable version

Tree style

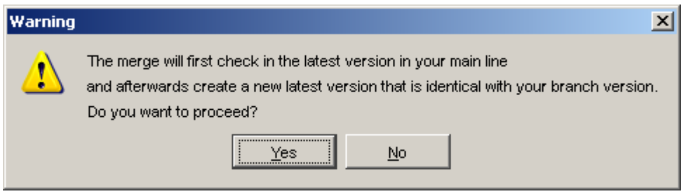
○ Versions only
○ Labels only
● Versions and Labels
○ ClearCase style

Order

● ascending
○ descending

Figure 232 - CM, Branch

After working on a branch, it is necessary sometimes to **merge** the results of a branch version (e.g.: V2.2.1 Branch) to the latest version of the main line (e.g.: V3 latest). There are 2 possibilities of merge support in TEMPPO. On one hand you can merge a whole test structure to the latest version of the main line, whereas the latest version will be checked-in before copying the branch. The additional version of the main line is necessary in order to keep the possibility of recovering the test structure of the main line. When pressing the merge button, the following question will appear.



Warning

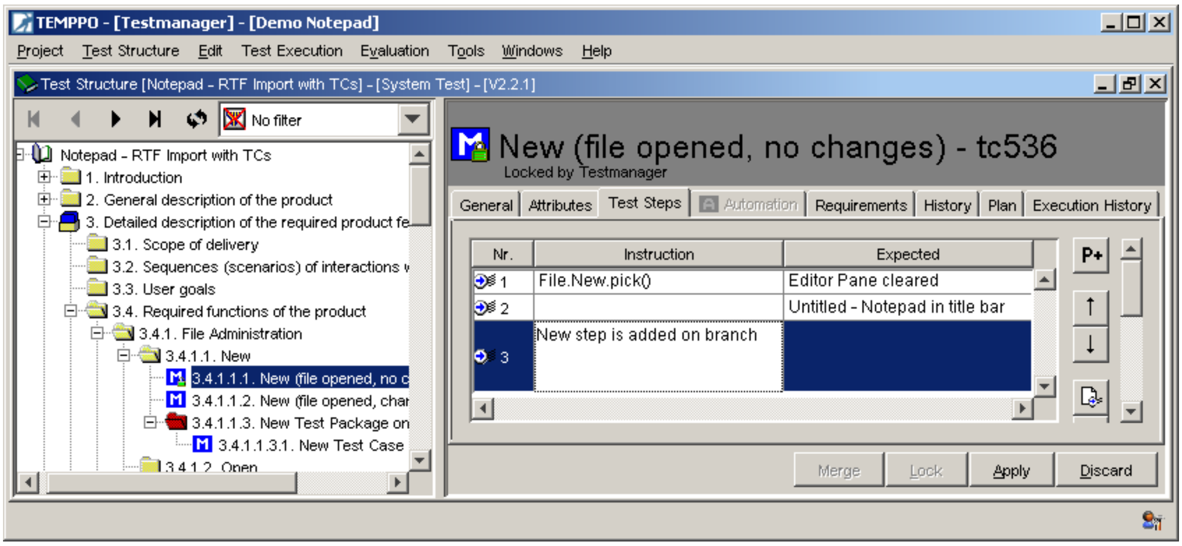
The merge will first check in the latest version in your main line
and afterwards create a new latest version that is identical with your branch version.
Do you want to proceed?

Figure 233 - Merge whole test structure

If you proceed, the version of the current branch version is “copied” to lastest version of the main line. The current version applied label remains activated.

Currently there is no visual representation for merges in the version dialog, like the merge arrows you would see in other CM tools.

On the other hand there is an individual merge that allows merging single test cases.

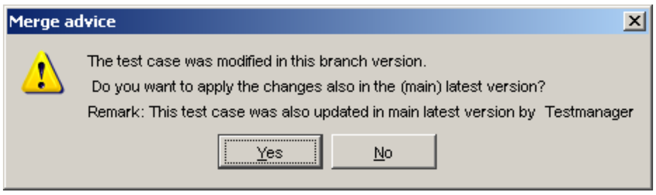

TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V2.2.1]
No filter
Notepad - RTF Import with TCs
1. Introduction
2. General description of the product
3. Detailed description of the required product fe
3.1. Scope of delivery
3.2. Sequences (scenarios) of interactions w
3.3. User goals
3.4. Required functions of the product
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file opened, no c
3.4.1.1.2. New (file opened, char
3.4.1.1.3. New Test Package on
3.4.1.1.3.1. New Test Case
3.4.1.2. Open
New (file opened, no changes) - tc536
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
P+
Merge Lock Apply Discard

Nr.	Instruction	Expected
1	File.New.pick()	Editor Pane cleared
2		Untitled - Notepad in title bar
3	New step is added on branch	

Figure 234 Test case merge

If working on a branch an additional button **Merge** appears besides Lock, Apply and Discard

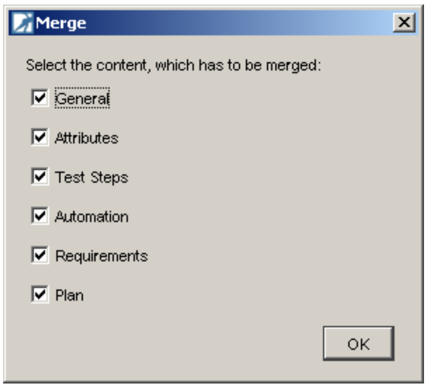
the current test case. After pressing the button Apply the user is automatically asked if he wants to apply the changes on the latest version (V3). Additionally the message shows the information, if the same test case was also changed in the latest version!



The test case was modified in this branch version.
Do you want to apply the changes also in the (main) latest version?
Remark: This test case was also updated in main latest version by Testmanager

Figure 235 - Merge advice

If the user presses **Yes**, the following window appears, where the user can select which parts of the test case will be merged.



Select the content, which has to be merged:

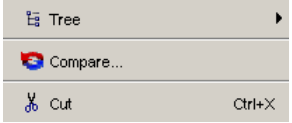
☑ General
☑ Attributes
☑ Test Steps
☑ Automation
☑ Requirements
☑ Plan

Figure 236 - Merge selection

If nothing is selected, test case is latest version remains unchanged!

3.3.12Compare

Version management also includes comparing a test case with a another one. If you select a test case in the current version of a test structure and activate the menu item **Compare**



a version window comes up where you can select the version of the test case to be compared. The versions window does only show the versions in which the test case exists. After selecting a version the button “Compare” gets enabled.

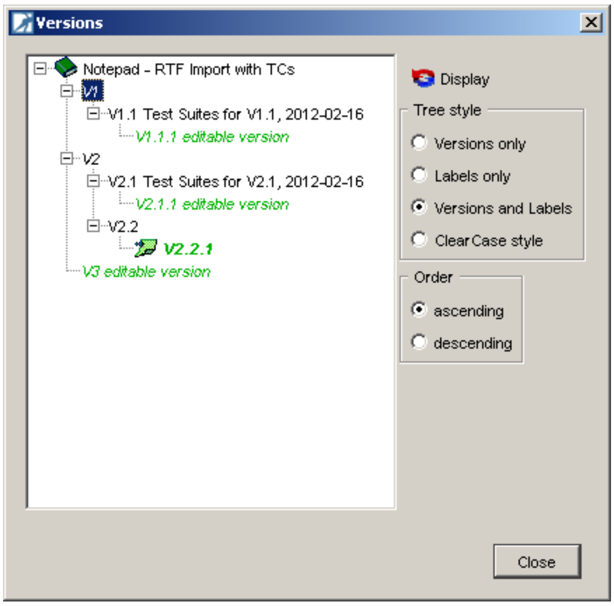
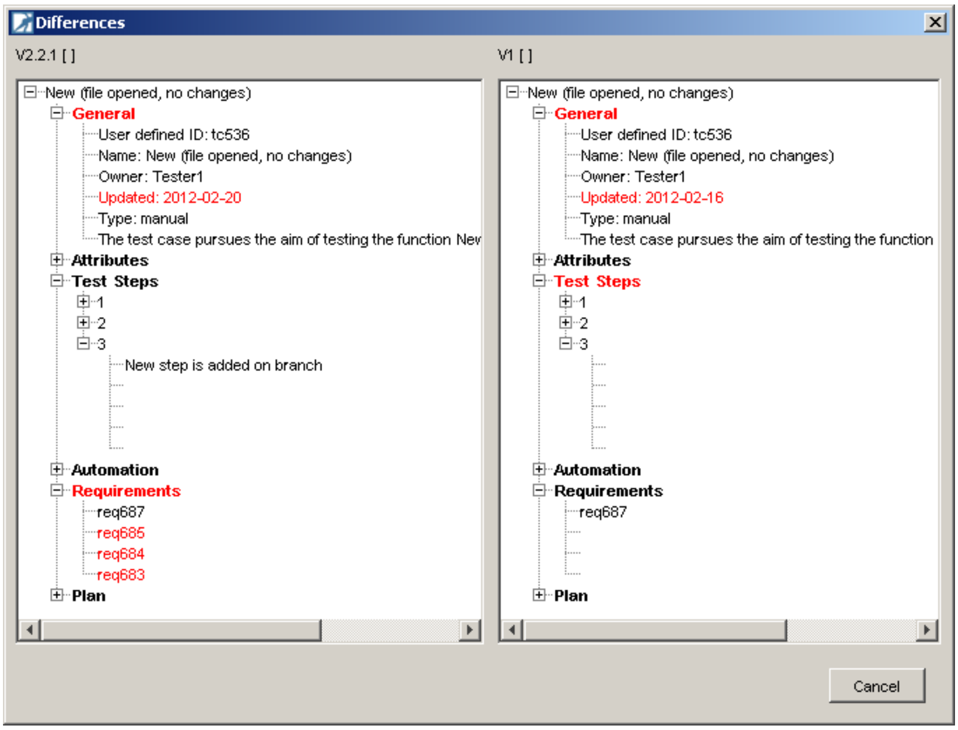


Figure 237 - Compare - selecting a version

When pressing the button “Compare”, Figure 238 is shown. The two versions of the test case are displayed with their differences in red color.



V2.2.1 [ ]

```
New (file opened, no changes)
  General
    User defined ID: tc536
    Name: New (file opened, no changes)
    Owner: Tester1
    Updated: 2012-02-20
    Type: manual
    The test case pursues the aim of testing the function Nev
  Attributes
  Test Steps
    1
    2
    3
      New step is added on branch
  Automation
  Requirements
    req687
    req685
    req684
    req683
  Plan
```

V1 [ ]

```
New (file opened, no changes)
  General
    User defined ID: tc536
    Name: New (file opened, no changes)
    Owner: Tester1
    Updated: 2012-02-16
    Type: manual
    The test case pursues the aim of testing the function
  Attributes
  Test Steps
    1
    2
    3
  Automation
  Requirements
    req687
  Plan
```

Figure 238 - Compare- differences

3.3.13Filter

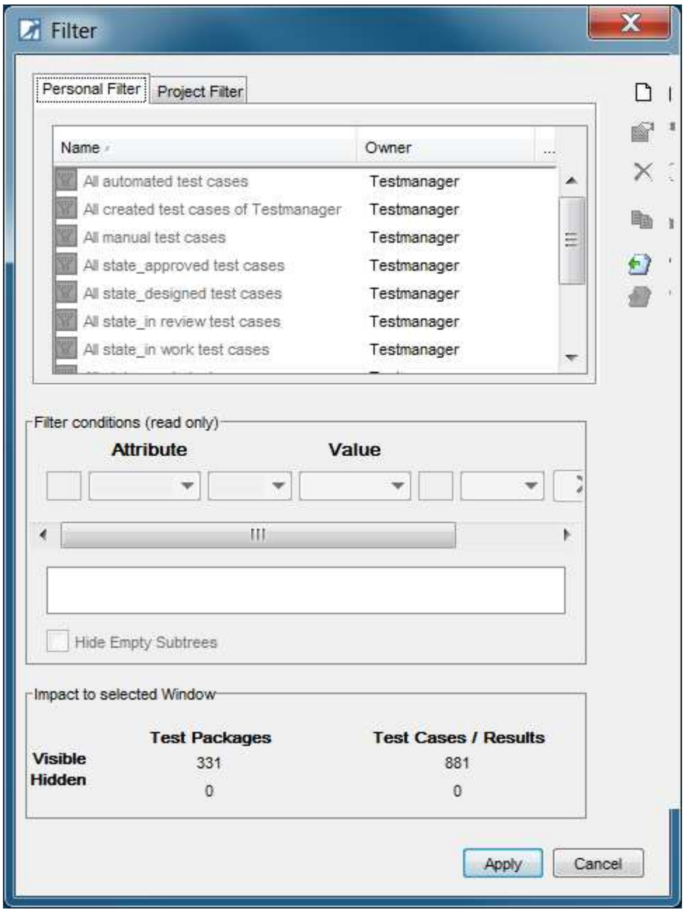
Users can see their test structures and test suites in a filtered way by creating and selecting filters i.e. for the purpose of hiding test cases that are not of interest. All fixed and user-defined attributes can be used to define a filter criterion.

Test structure filters are available for test suites. Test Suite filters which are including result or tester are not available for test structures.

TEMPPO Manager offers **Personal** and **Project filters**. If the tab **Personal Filter** is activated, all created filters are only displayed for the user. A test manager may activate the tab **Project Filter** and creates filters that can be used for all project members. **Personal filters** can be copied to **project** ones and vice versa.

3.3.13.1 Predefined filters

There are a lot of predefined filter which are useful for each user.



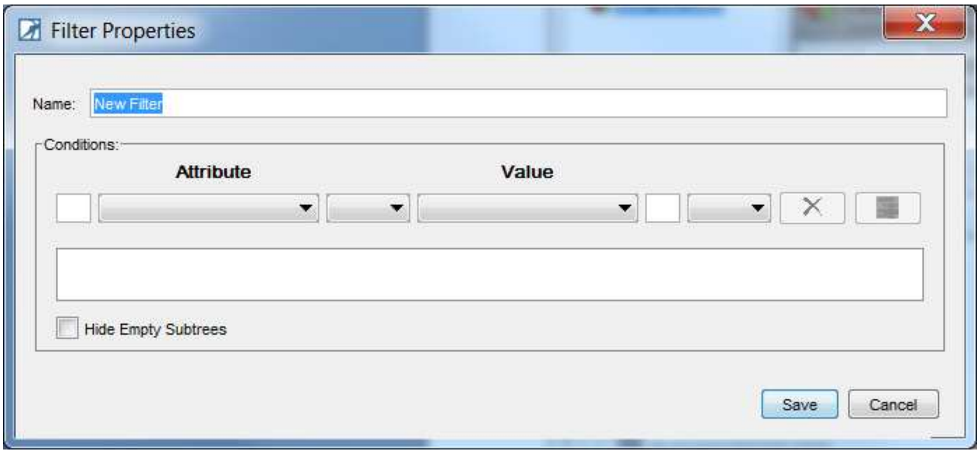
Name	Owner
All automated test cases	Testmanager
All created test cases of Testmanager	Testmanager
All manual test cases	Testmanager
All state_approved test cases	Testmanager
All state_designed test cases	Testmanager
All state_in review test cases	Testmanager
All state_in work test cases	Testmanager

	Test Packages	Test Cases / Results
Visible	331	881
Hidden	0	0

Figure 239 - Predefined filters

3.3.13.2 Create filter

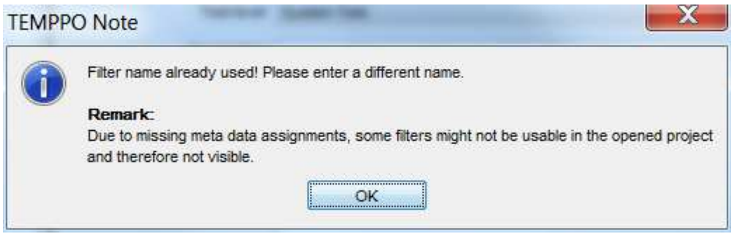
You can create a new filter when pressing the button **New filter** in Figure 239.



Name: New Filter

Figure 240 - Filter Properties

Pressing the **Save**-Button, the new filter is saved to the database, if a new name was entered. Otherwise the information window comes up:



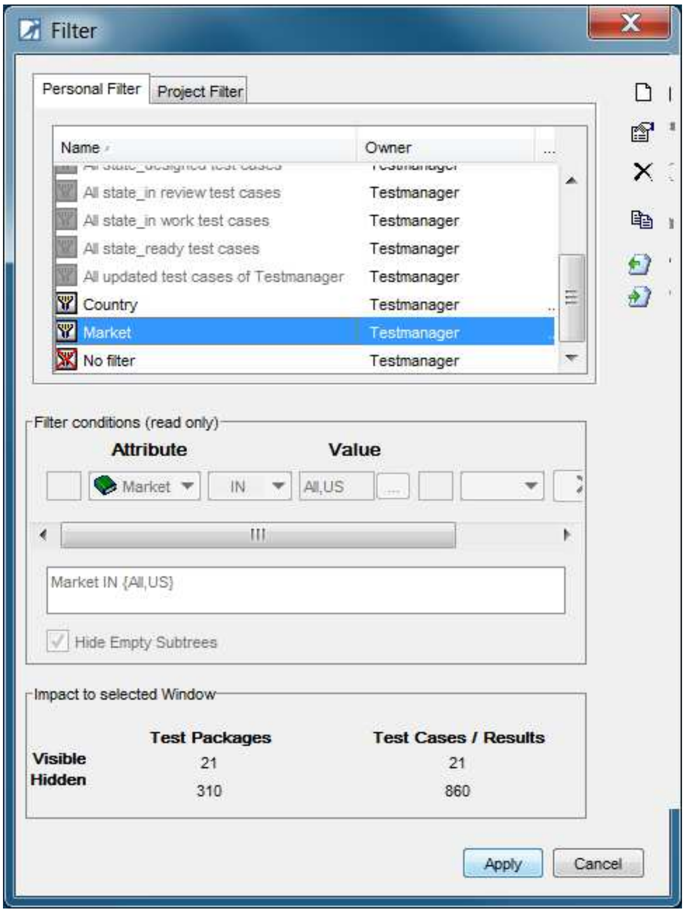
TEMPPO Note

Filter name already used! Please enter a different name.

Remark:
Due to missing meta data assignments, some filters might not be usable in the opened project and therefore not visible.

Figure 241 - Filter name already used

After saving the new filter, Figure 242is shown. The new filter is inserted and selected in the list box, the filter criterion is displayed, and the table **Impact to selected Window** is updated.



Name	Owner
All state_in review test cases	Testmanager
All state_in work test cases	Testmanager
All state_ready test cases	Testmanager
All updated test cases of Testmanager	Testmanager
Country	Testmanager
Market	Testmanager
No filter	Testmanager

Filter conditions (read only)

Attribute		Value
Market	IN	All,US

Market IN {All,US}

☑ Hide Empty Subtrees

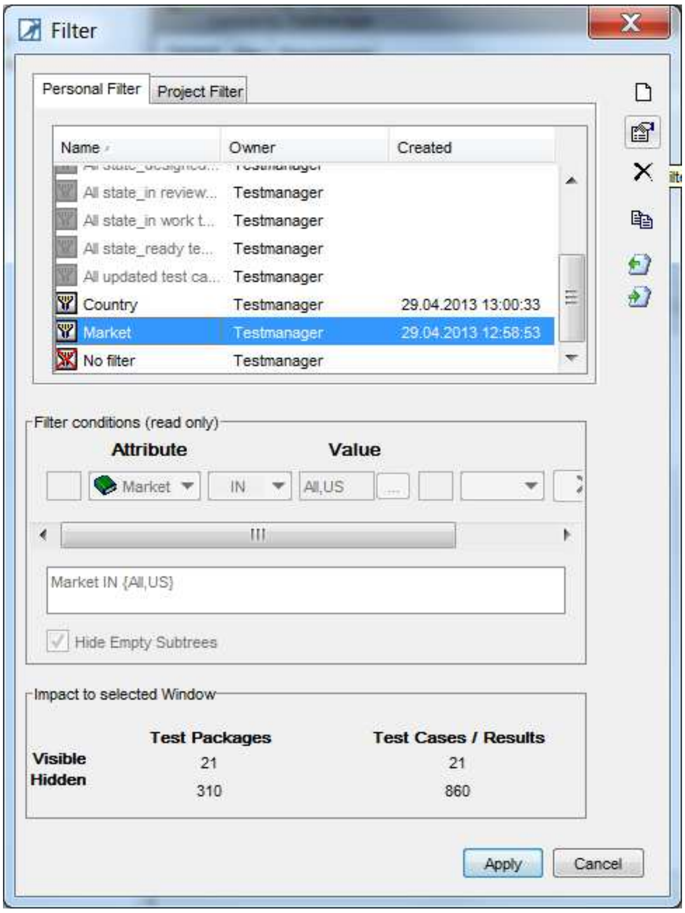
Impact to selected Window

	Test Packages	Test Cases / Results
Visible	21	21
Hidden	310	860

Figure 242 - Selected filter

3.3.13.3 Change filter

You can change existing filters by pressing the **Edit filter** button. This button is only enabled when a filter is selected that is not the empty filter (**No filter**).



Name	Owner	Created
All state_designed...	Testmanager	
All state_in review...	Testmanager	
All state_in work t...	Testmanager	
All state_ready te...	Testmanager	
All updated test ca...	Testmanager	
Country	Testmanager	29.04.2013 13:00:33
Market	Testmanager	29.04.2013 12:58:53
No filter	Testmanager	

Filter conditions (read only)

Attribute		Value
Market	IN	All,US

Market IN {All,US}

☑ Hide Empty Subtrees

Impact to selected Window

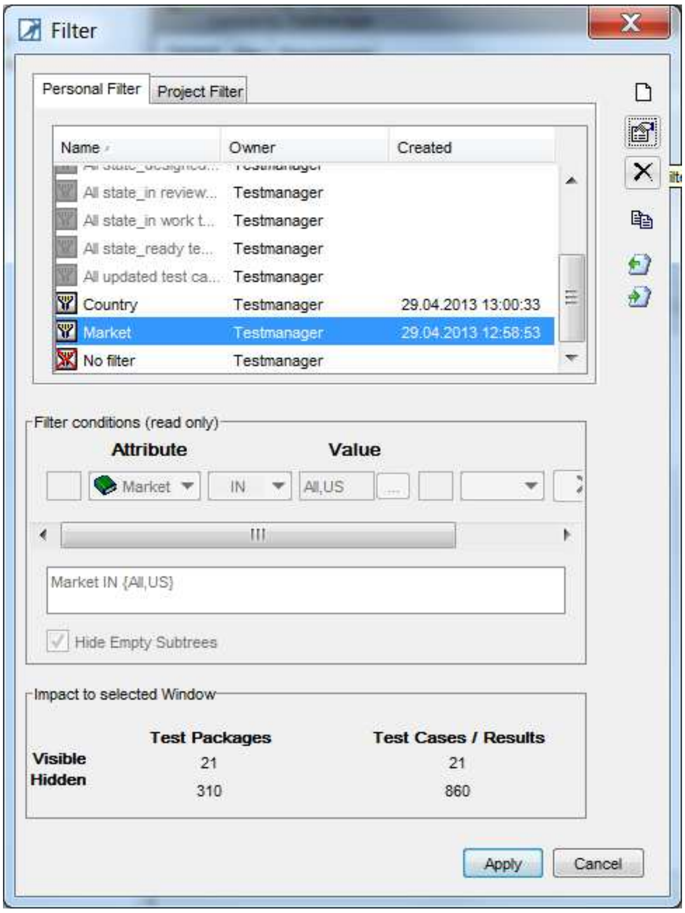
	Test Packages	Test Cases / Results
Visible	21	21
Hidden	310	860

Figure 243 – Edit filter

Depending on the selected filter, the conditions are displayed. Filter name is read only, filter conditions can be changed. (See Figure 240)

3.3.13.4 Delete filter

You can delete an existing filter by pressing the **Delete filter** -button. This button is only enabled if the selected filter is not the empty filter (**No filter**).



Filter

Personal Filter | Project Filter

Name	Owner	Created
All state_designed...	Testmanager	
All state_in review...	Testmanager	
All state_in work t...	Testmanager	
All state_ready te...	Testmanager	
All updated test ca...	Testmanager	
Country	Testmanager	29.04.2013 13:00:33
Market	Testmanager	29.04.2013 12:58:53
No filter	Testmanager	

Filter conditions (read only)

Attribute		Value	
Market	IN	All,US	...

Market IN {All,US}

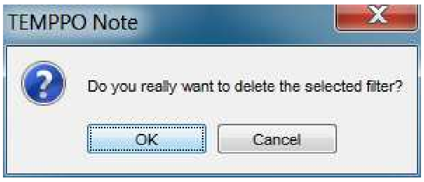
☑ Hide Empty Subtrees

Impact to selected Window

	Test Packages	Test Cases / Results
Visible	21	21
Hidden	310	860

Figure 244 - Delete Filter

After activating the **Delete filter** -button, the reconfirmation window comes up:

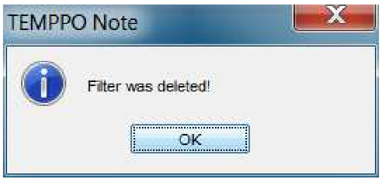


TEMPPO Note

Do you really want to delete the selected filter?

Figure 245 - Delete filter

If pressing **Yes**, the information window comes up and the filter is deleted, if it is not used by a test structure or suite.



TEMPPO Note

Filter was deleted!

Figure 246 - Filter deleted

After deleting a filter, the **No filter** is highlighted in the list box.

3.3.13.5 Defining the filter criterion

The values of the following attributes can be used as described in Table 2:

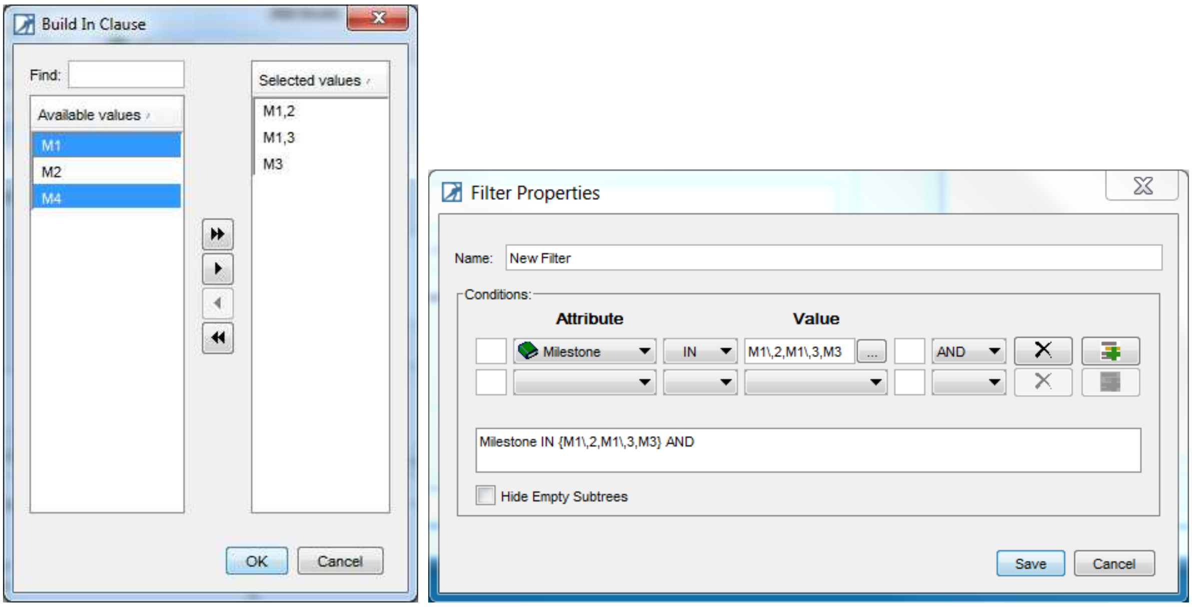
Fixed[4], UDAs, test case name, owner, tester, bug ID, test result, requirements

First of all you have to choose an attribute and an operator (**=, !=, IN, NOT IN**). If **IN** or **NOT IN** is selected, the combo-box value changes to a text field together with the button "**...**". There you can enter

- A string with a wild card or
- Press the button "..." for selecting several values.

After pressing the button "**...**" Figure 247 is displayed and the attribute values are selected by moving them to the right. Pressing **OK** applies the selection to the filter window. The selected values are displayed separated by a comma. Now the user can edit this line. Because the comma is used to separate the selected values, commas in the selected values have to be escaped with a "\" (e.g. Component\,3).

ⓘ **No blank is allowed, commas has to be escaped with a "\"!**



Name: New Filter

Conditions:

Milestone IN M1\,2,M1\,3,M3 ... AND

Milestone IN {M1\,2,M1\,3,M3} AND

Figure 247 - Selecting attribute values for IN, NOT IN

Attributes	Operator =, !=	Operator IN, NOT IN	Operator <,>, <=, >=	Value editable[1]	Value set[2]	Value project assigned	Wild card allowed[3]	Test Structure
Fixed I[4]	Y	N	N	N	N	-	N	Y
Fixed II[5]	Y	Y	N	Y	Y	-	Y	Y
Fixed III[6]	Y	Y	N	N	Y	-	N	Y
UDA	Y	Y	N	Y	Y	Y	Y	Y

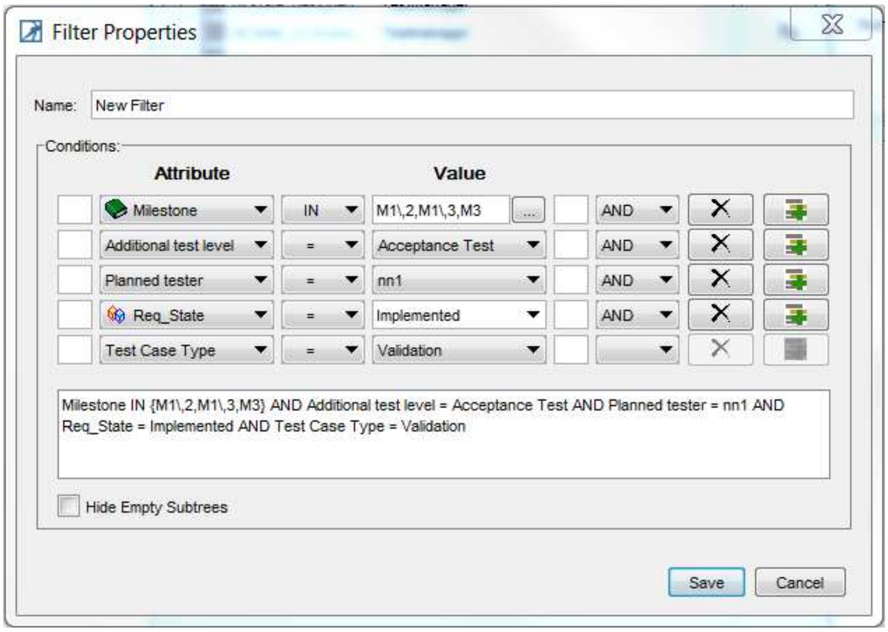
[1] Yes, if operators IN, NOT IN are selected
[2] Yes, if a set of values can be selected
[3] Yes, if editable is Yes.
[4] Fixed I: Situation, Test Case Type, Additional Test Level, Type
[5] Fixed II: Test Level, Test Category, Requirement
[6] Fixed III: Priority, State

TC name	Y	N	N	Y	N	-	Y	Y
Owner	Y	Y	N	Y	Y	Y	Y	Y
Tester	Y	Y	N	Y	Y	Y	Y	N
BugID	Y	Y	N	Y	Y	-	Y	N
Updater	Y	N	N	N	N	-	N	Y
Test result	Y	Y	N	N	N	-	N	N
Created, Updated	Y	N	Y	N	N	-	-	Y
Requirement	Y	Y	N	Y	Y	Y	Y	N

Table 2- Attributes meanings

The user can write free text that may also contain wildcards like

"*"	any number of characters
"?"	any single character



Filter Properties

Name: New Filter

Conditions:

	Attribute		Value				
□	Milestone	IN	M1\,2,M1\,3,M3 ...	□	AND	✕	➕
□	Additional test level	=	Acceptance Test	□	AND	✕	➕
□	Planned tester	=	nn1	□	AND	✕	➕
□	Req_State	=	Implemented	□	AND	✕	➕
□	Test Case Type	=	Validation	□		✕	

Milestone IN {M1\,2,M1\,3,M3} AND Additional test level = Acceptance Test AND Planned tester = nn1 AND Req_State = Implemented AND Test Case Type = Validation

□ Hide Empty Subtrees

Figure 248 - Filter criterion

3.3.13.6 Edit filter criterion

You delete and add lines of filter criterion.

When pressing the button

- ✕ the current line is deleted.
- ➕ a line is inserted before the current one.

3.3.13.7 Export filter

It is possible to export filters to XML and import them again. Via (multi) selection of filter and pressing the button **Export filters to XML** filter are saved to an xml file.

3.3.13.8 Import filter

It is possible to import filters from XML by pressing the button **Import filter from xml**. A filter import is rejected, if the filter contains specific attributes that are not assigned to the project the filter is imported to. An error message is displayed where the names of the NOT imported filters are listed.

The filter contents are checked and the import is only successful, if the attributes match the ones of the selected project. If the filter is not syntactically correct, the import is cancelled and the user gets an information message.

If an XML file contains a test suite filter and it is imported to a test structure (or vice versa), the corresponding test suite filter is imported and an information window is shown. The imported test suite filter is shown the first time when a test suite is opened and the filter dialog is called (or the filter combo box is clicked).

3.3.14Test structure XML Export

This feature allows exporting the entire test structure or just a part of it into an XML document. When a test structure is open and the root node or a test package is selected, the menu item **Test Structure – Export Structure…** is enabled.

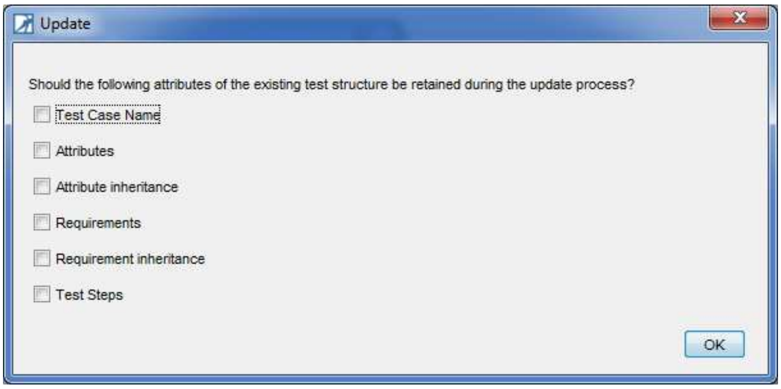
On activating this menu item a file open dialog is displayed. When pressing the button `Save` the test structure is saved as XML file based on **teststructure.dtd.**

3.3.15Test structure XML Update

This feature allows updating the entire test structure or test packages based on an XML document, if you are using other tools for manipulating the test structure. The update process adapts the test structure to XML file, but keeps TC ids and TP ids and real updates of TCs and TPs are detected.

If you activating the menu item **Test structure -> Update -> XML document**, a file open dialog comes up where you can select the XML file. After pressing the button Open, the XML document is read and checked against **teststructure.dtd**. If an error is detected, the import process is cancelled.

If the syntax was checked successfully, you can specify to retain some TP/TC properties like test case name, attributes, attribute inheritance, requirement inheritance, test steps, which will be kept after the update process. Only the other content is updated.



Should the following attributes of the existing test structure be retained during the update process?

□ Test Case Name

□ Attributes

□ Attribute inheritance

□ Requirements

□ Requirement inheritance

□ Test Steps

Figure 249 – Select properties to be retained

After pressing **OK** the test structure is checked of any user locks. If a user has locked an item the test structure can't be checked-in and the update process is cancelled with a corresponding message.

If there aren't any locks the test structure is checked-in with label "**Checked in by XML update**" and the new latest version is completely locked during the update process.

> ⓘ **If you update only a part of a test structure, the selected TP and root TP of XML have to be the same, otherwise the update is not started.**

3.3.15.1.1 Update process

The update process is done as follows: The XML structure becomes the new test structure. Each item is recursively compared with itself in the latest test structure version by its ID.

If the item has been changed, the changes are applied and "**Updated**" and "**Updator**" are written. History is written like current behavior. If a mandatory history comment is configured for that project, it will be ignored.

If the item hasn't been changed, nothing happens.

Reorder: If an item is "reordered", it keeps the new order, but neither "Updated" nor "Updator" is written.

Changed TCs: Detected changes: Name, test goal, attributes, planning attributes, fields for automated TCs, new attribute assignments, deleted attribute assignments, new requirement assignments, and deleted requirement assignments.

If a change in test steps is detected, all steps of XML file are applied.

Changed TPs: Detected changes: Name, description, test category change (new, delete, change), document path, document name, document chapter, new attribute assignments, deleted attribute assignments, new requirement assignments, and deleted requirement assignments.

New TPs: If a new TP is detected, whose ID doesn't exist in test structure, the TP is created. No history entry is created (like a "create" on the GUI).

New TCs: If a TC is detected, whose ID doesn't exist in test structure, the TC is created. No history entry is created (like a "create" on the GUI).

3.3.16Test Suite XML Export / Import

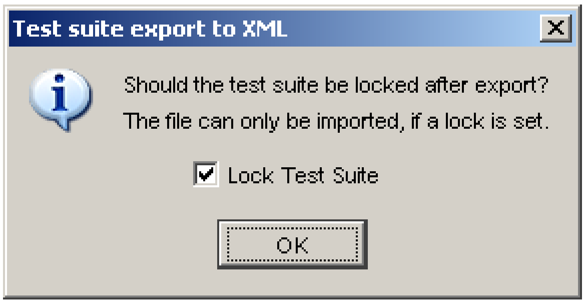
This feature allows exporting the entire or a part of a test suite into an XML document, fill in some test execution data and import that information to TEMPPO. E.g. some users may have complicated test automation tools/scenarios whose test cases can't be started and executed from TEMPPO but are managed in TEMPPO.

Therefore those test cases are administrated in TEMPPO, and after executing them outside of TEMPPO the results are imported.

3.3.16.1 Export

The export XML feature can either be used to execute automated TCs somewhere offline and export results later or export the test suite to an XML file for other matters. In this case the test suite shouldn't be locked.

A test suite can be exported if a test suite is already opened, any node (test suite root or TP) is selected and the menu item **Export test suite -> to XML** is activated.



Should the test suite be locked after export?

The file can only be imported, if a lock is set.

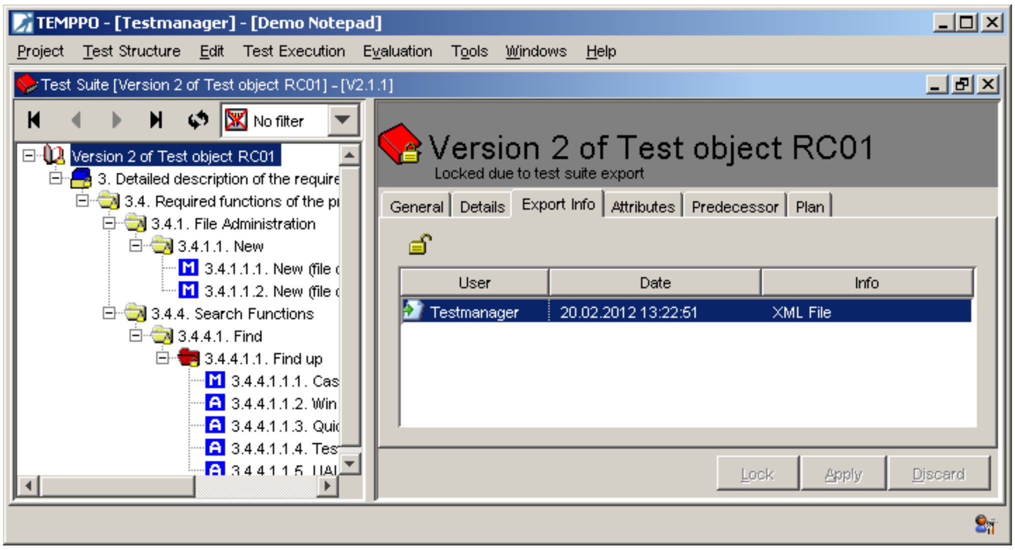
☑ Lock Test Suite

Figure 250 - Lock confirmation

After entering a **name**, the XML file is stored to the file system (conforming to **testsuite.dtd**).

ⓘ **The testsuite.dtd can be found in your TEMPPO/XML directory**

After exporting a test suite to an XML file, all exports (also to the export DB) are listed in the **Export Info**.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Suite [Version 2 of Test object RC01] - [V2.1.1]
No filter
Version 2 of Test object RC01
3. Detailed description of the require
3.4. Required functions of the pi
3.4.1. File Administration
3.4.1.1. New
3.4.1.1.1. New (file c
3.4.1.1.2. New (file c
3.4.4. Search Functions
3.4.4.1. Find
3.4.4.1.1. Find up
3.4.4.1.1.1. Cas
3.4.4.1.1.2. Win
3.4.4.1.1.3. Quic
3.4.4.1.1.4. Tes
3.4.4.1.1.5. UAI
General Details Export Info Attributes Predecessor Plan
Lock Apply Discard

Version 2 of Test object RC01

Locked due to test suite export

User	Date	Info
Testmanager	20.02.2012 13:22:51	XML File

Figure 251 – Export XML information

TEMPPO identifies an XML file by a unique ID.

3.3.16.2 Import

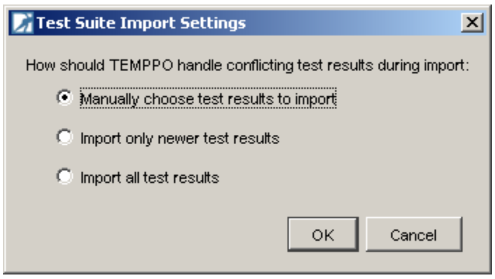
After exporting an XML file, it may be edited outside of TEMPPO, e.g. users or test automation tools are filling in some values. Valid values are:

- **Result** = (Failed | Passed | Not_executed | Blocked | Not_completed | Not_implemented | On_hold)
- **Result Output** = Plain text
- **Tester** = (MS Windows account or a defined user account)
- **Tested** = DDMMYY HH:MM:SS

If other values are changed, they will be ignored.

Importing an XML file can be started by activating the menu item **Import Test suite -> from XML**. The import is independent from the selected node. TEMPPO detects the part to be imported.

The following possibilities are offered for importing:



How should TEMPPO handle conflicting test results during import:

● Manually choose test results to import

○ Import only newer test results

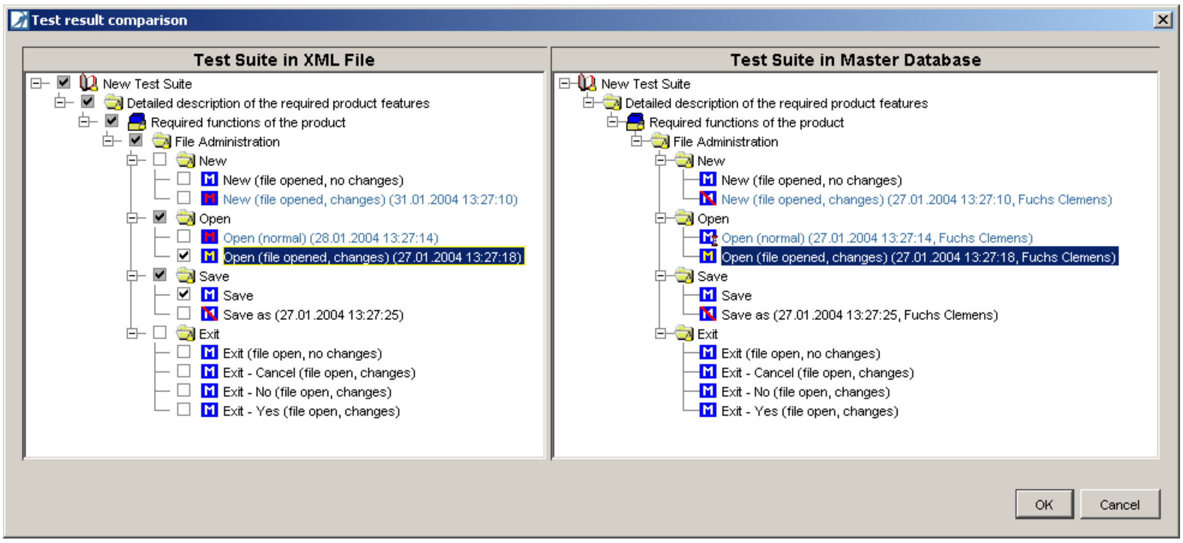
○ Import all test results

Figure 252 – XML Import

- **Manually choose test results to import**: When activating this option, Figure 253 is shown. In the right part of the window the current (result) state of test cases in the central database is shown together with their execution date (NOT importing date!). Test cases that may cause conflicts are colored blue. The left side of the window shows the test results of the exported test suite. Now you can activate the checkboxes of test cases that you want to import.

- **Import only newer test results**: If a user already imported a test result for an automated test case and you also executed the same test case
    - Earlier, the result in the central database won't be overridden by yours.
    - Later, your result will be written into the database. The other result is lost.
- **Import all test results**: All test results of already executed test cases are overridden by your results

ⓘ **Note: All test results that are marked are overridden in the central database after closing the window with OK.**



Test Suite in XML File

- ☑ New Test Suite
    - ☑ Detailed description of the required product features
        - ☑ Required functions of the product
            - ☑ File Administration
                - ☐ New
                    - ☐ New (file opened, no changes)
                    - ☐ New (file opened, changes) (31.01.2004 13:27:10)
                - ☑ Open
                    - ☐ Open (normal) (28.01.2004 13:27:14)
                    - ☑ Open (file opened, changes) (27.01.2004 13:27:18)
                - ☑ Save
                    - ☑ Save
                    - ☐ Save as (27.01.2004 13:27:25)
                - ☐ Exit
                    - ☐ Exit (file open, no changes)
                    - ☐ Exit - Cancel (file open, changes)
                    - ☐ Exit - No (file open, changes)
                    - ☐ Exit - Yes (file open, changes)

Test Suite in Master Database

- New Test Suite
    - Detailed description of the required product features
        - Required functions of the product
            - File Administration
                - New
                    - New (file opened, no changes)
                    - New (file opened, changes) (27.01.2004 13:27:10, Fuchs Clemens)
                - Open
                    - Open (normal) (27.01.2004 13:27:14, Fuchs Clemens)
                    - Open (file opened, changes) (27.01.2004 13:27:18, Fuchs Clemens)
                - Save
                    - Save
                    - Save as (27.01.2004 13:27:25, Fuchs Clemens)
                - Exit
                    - Exit (file open, no changes)
                    - Exit - Cancel (file open, changes)
                    - Exit - No (file open, changes)
                    - Exit - Yes (file open, changes)

Figure 253 – Manual import of test results

Rules for importing:

If test results differ from test step results (e.g. test case result = passed, test step 1 result = failed, test step 2 result = passed), the new test case result will be failed.

If a test case has already a result, result output and it is set to not executed, tester and tested will be deleted (=empty) and result output is kept or updated.

3.3.16.2.1 Import characters of other fonts

If you want to import Chinese or Arabic characters the xml header has to be

<?xml version="1.0" encoding="**UTF-8**"?>

For more information see http://de.wikipedia.org/wiki/UTF-8 and http://www.utf-8.com/

3.3.17Export Test Suite to DOORS

This feature allows exporting Test Suite Results – of test cases where requirements from DOORS Requirement Structures are linked – to DOORS.

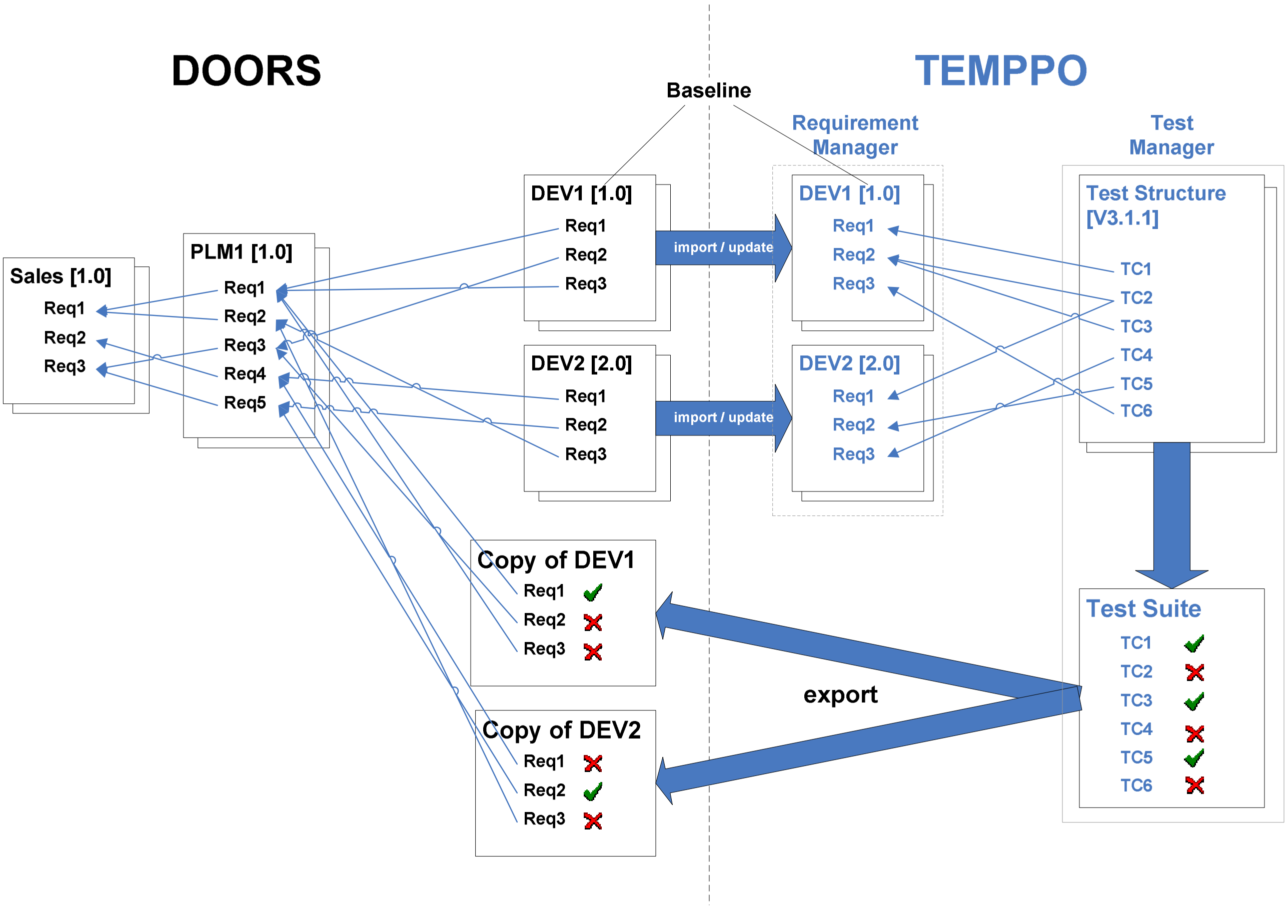
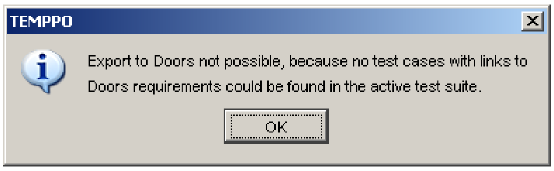


Figure 254 – DOORS – TEMPPO data exchange

The module Sales specifies requirements, which are refined in the module PLM1, which for their part are refined in the modules DEV1 and DEV2. DEV1 and DEV2 (certain baseline or latest version) are imported into TEMPPO and provide a base for the test cases organized in a test structure. For test execution a test suite is created, where test results are recorded, which can be reflected onto the linked requirements. When exporting the test results to DOORS, the imported modules are copied (DEV1 and DEV2) and the requirements in the copies will be flagged with test results. Since a single requirement can be covered by several test cases (Req2 in DEV1), rules are applied, to build unique results for requirements (e.g. passed + failed = failed).

Test Suite results can be exported, if a test suite is open and active and the menu item **Export test suite -> to DOORS** is activated.

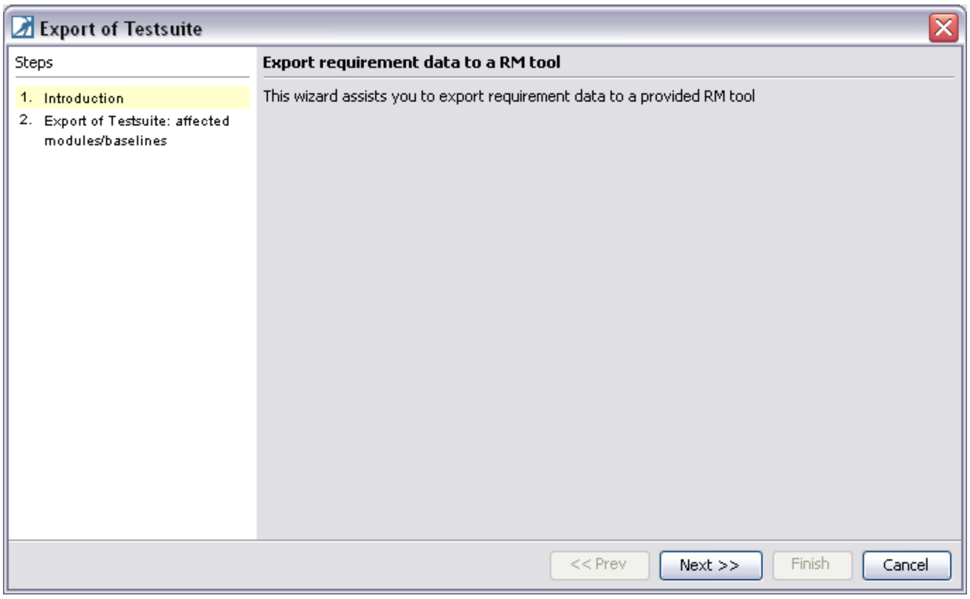
TEMPPO first verifies, whether the test suite contains any test cases linked to requirements imported from DOORS. If no linked requirements can be found, the user is informed via a message and the process is aborted:


TEMPPO

Export to Doors not possible, because no test cases with links to Doors requirements could be found in the active test suite.

Figure 255 – Information Message

If there are linked requirements, the user is guided through the wizard.

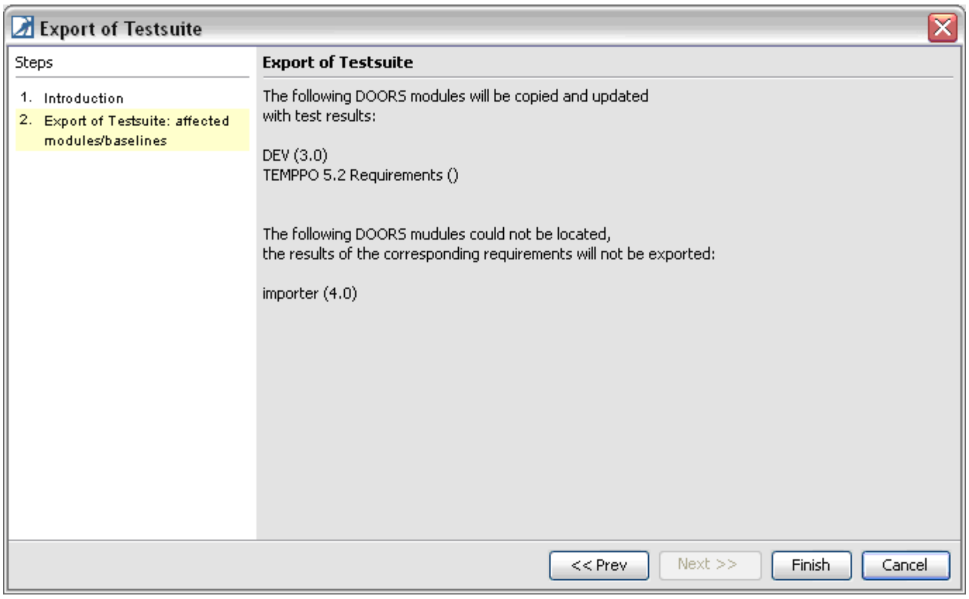


Export requirement data to a RM tool

This wizard assists you to export requirement data to a provided RM tool

Figure 256 – Export Wizard – step 1

TEMPPO starts the communication with DOORS and searches for the affected module(s) / baseline(s) in the DOORS database. Before starting the export, TEMPPO gives an overview:



Export of Testsuite

The following DOORS modules will be copied and updated
with test results:

DEV (3.0)
TEMPPO 5.2 Requirements ()

The following DOORS mudules could not be located,
the results of the corresponding requirements will not be exported:

importer (4.0)

Figure 257 – Export Wizard – step 2

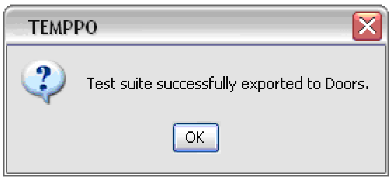
Results for found modules will be exported; results for not found modules will be skipped.

If a requirement is covered by more than one test case, a unique result will be built according to following rules (for description of test results please refer to 3.1.4.1.5):

- **not executed**, if all test cases are not executed
- **passed**, if all test cases are passed
- **failed**, if at least one test case is failed
- **blocked**, if no test case is failed and at least one is blocked

- **not completed**, if at least one test case is passed and all others are not executed
- **not implemented**, if at least one test case is not implemented (stronger than “on hold”)
- **on hold**, if at least one test case is on hold

When the export is finished, the user will be informed via message box.

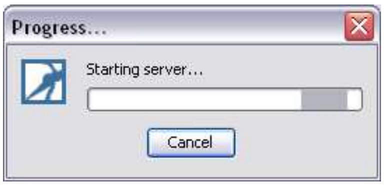


TEMPPO

Test suite successfully exported to Doors.

Figure 258 – Message box – export finished

The Export can be canceled at any time by clicking the cancel button in the progress monitor.

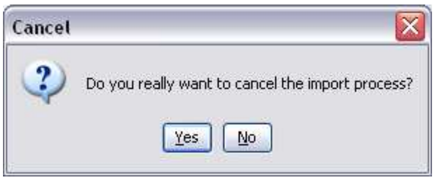


Progress...

Starting server...

Figure 259 – DOORS Export – progress monitor

The User is asked for confirmation then and if he confirms, the export is stopped.



Cancel

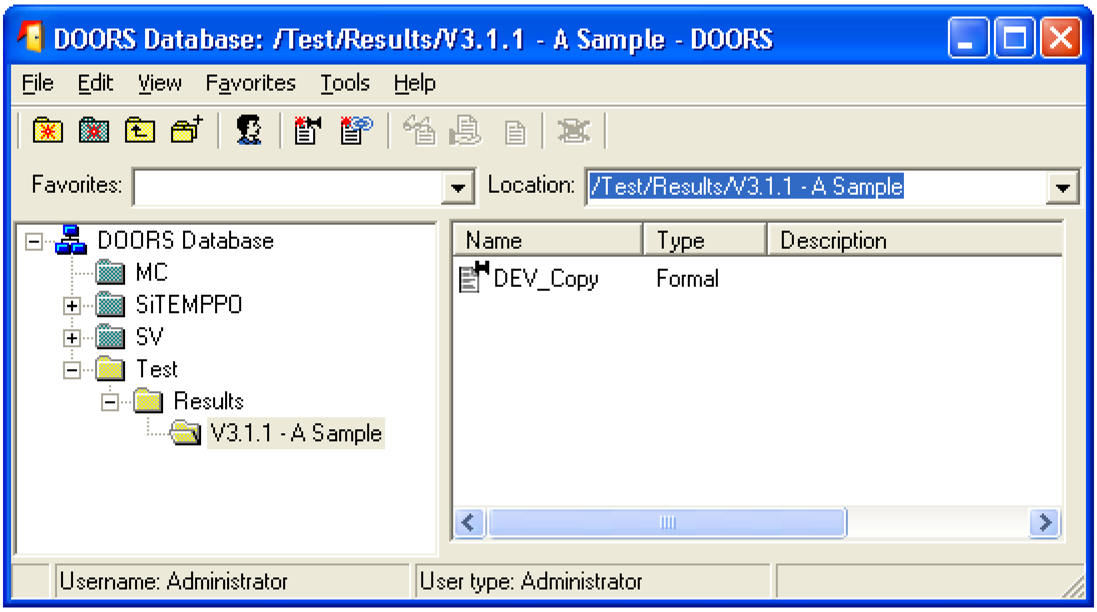
Do you really want to cancel the import process?

Figure 260 – Doors Export- confirm cancel.

3.3.17.1 Effects in DOORS

Following steps will be processed during export:

1. Creation of folder for test execution data (if not already existing) according to following naming convention:
   \Test\Results\<Version> - <Name of Test Suite>
2. Copy of affected DOORS modules (corresponding baseline or latest version) to test execution folder if copies do not already exist. The module is constructed by <subfolder 1>-<subfolder 2>-<subfolder n>-<module name>-<baseline name>



DOORS Database: /Test/Results/V3.1.1 - A Sample - DOORS

File Edit View Favorites Tools Help
Favorites:
Location: /Test/Results/V3.1.1 - A Sample
DOORS Database
MC
SiTEMPPO
SV
Test
Results
V3.1.1 - A Sample

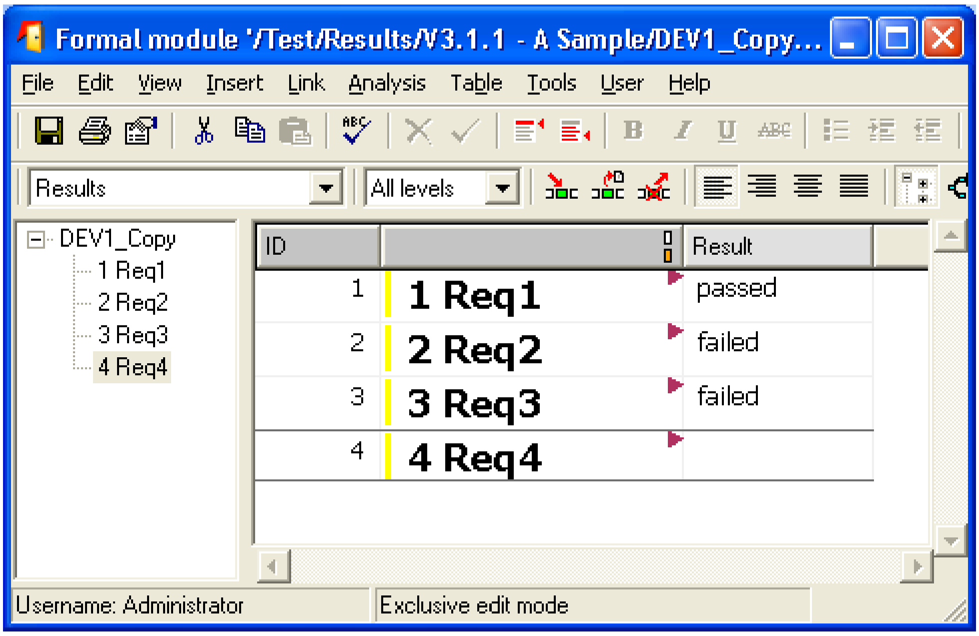
Name	Type	Description
DEV_Copy	Formal	

Username: Administrator | User type: Administrator

Figure 261: Copy of DEV module

3. Creation of module attribute “test result” in each module copy
   - Write test result into attribute “test result” of affected requirements. If a requirement is covered by more than one test case, a unique result will be built according to rules described above.

   The attribute “test result” of not affected requirements will remain empty.



Formal module '/Test/Results/V3.1.1 - A Sample/DEV1_Copy...

ID		Result
1	**1 Req1**	passed
2	**2 Req2**	failed
3	**3 Req3**	failed
4	**4 Req4**	

Username: Administrator | Exclusive edit mode

Figure 262: Results for requirements

3.3.18Test Suite Scheduler

Manual test cases are executed sequential, but executing automated test cases, it is insufficient to do it in a sequential way. Therefore TEMPPO provides a feature to schedule automated test cases.

A scheduler includes general information, a start time and an ordered selection of test cases which has to be executed.

For each test suite several schedulers can be defined. The test suite has to be opened and tab **"Scheduler"** has to be activated:

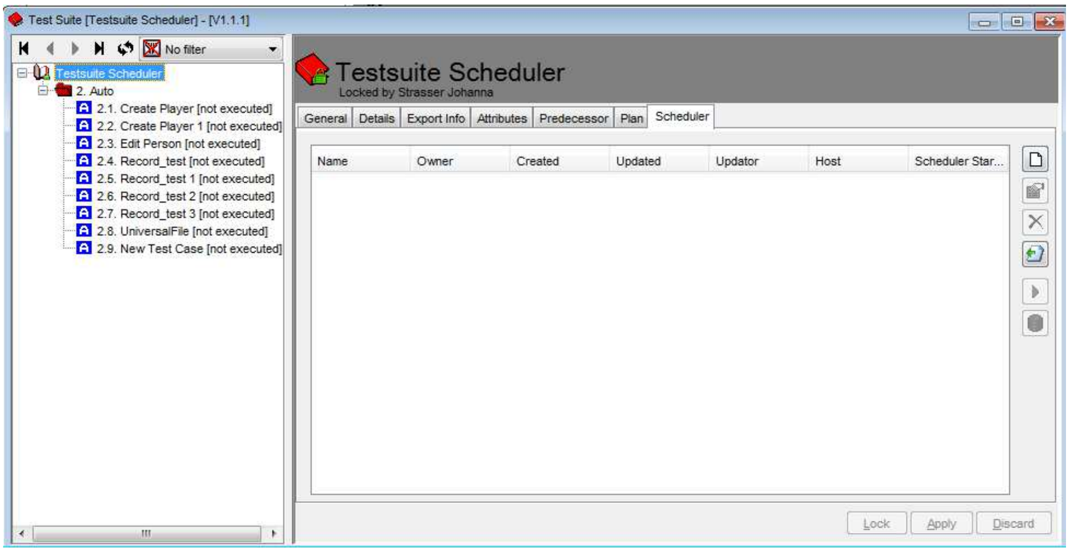

Test Suite [Testsuite Scheduler] - [V1.1.1]
No filter
Testsuite Scheduler
2. Auto
2.1. Create Player [not executed]
2.2. Create Player 1 [not executed]
2.3. Edit Person [not executed]
2.4. Record_test [not executed]
2.5. Record_test 1 [not executed]
2.6. Record_test 2 [not executed]
2.7. Record_test 3 [not executed]
2.8. UniversalFile [not executed]
2.9. New Test Case [not executed]
Testsuite Scheduler
Locked by Strasser Johanna
General
Details
Export Info
Attributes
Predecessor
Plan
Scheduler
Name
Owner
Created
Updated
Updator
Host
Scheduler Star...
Lock
Apply
Discard

Figure 263 – Test Suite Scheduler

With the buttons on the right side you can manage and run schedulers:

- **Create/Edit** a scheduler: Define a scheduler with start time and a sequence of test cases, see 3.3.18.1.
- **Delete** a scheduler: Remove a scheduler for a test suite, see 3.3.18.2.
- **Import** a scheduler: Import an existing scheduler from other test suites, see 3.3.18.3.
- **Run** a scheduler: Execute all test cases selected for scheduler, now or at start time, see 3.3.18.4.
- **Protocol** of a scheduler: Visit all results set by scheduler, see 3.3.18.5.

3.3.18.1 Create/Edit Scheduler

After opening a test suite, activating tab **Scheduler** and pressing button "New" / "Edit", new window opens where all settings for the scheduler can be defined:

In register **General:**

- **Name:** for scheduler
- **Scheduler Start Time:** Scheduler has to be started until this time. Scheduler cannot be started after this date.
- **Description:** for scheduler
- **Selection of test cases:** which has to be executed

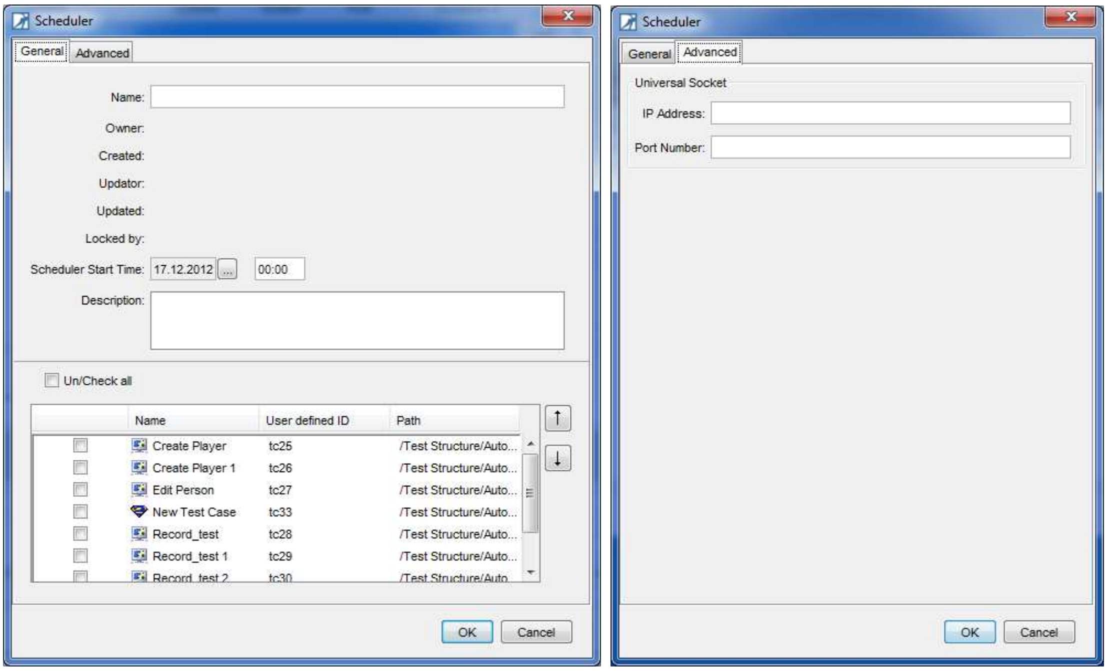
In register **Advanced:**

- **IP address, Port:** An IP address and port can be defined for all automated test cases using Universal Socket.

Remark: Selection of test cases:

All test cases in tree are shown in a list with a checkbox. Only the selected test cases are executed when scheduler is started.

The execution of test cases can be ordered with the buttons beside the table.



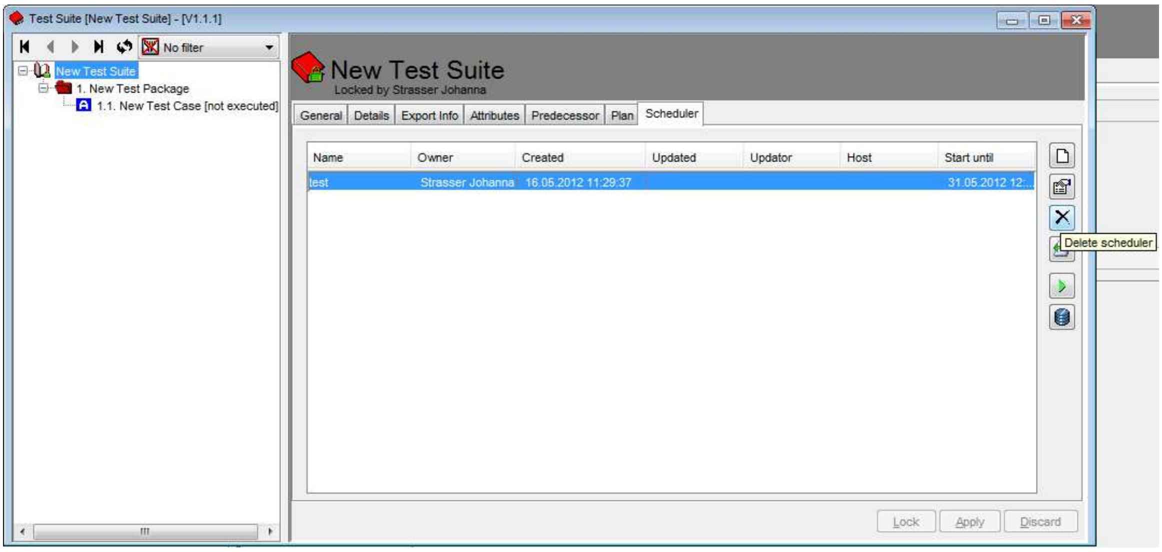
Scheduler Start Time: 17.12.2012 00:00

	Name	User defined ID	Path
□	Create Player	tc25	/Test Structure/Auto...
□	Create Player 1	tc26	/Test Structure/Auto...
□	Edit Person	tc27	/Test Structure/Auto...
□	New Test Case	tc33	/Test Structure/Auto...
□	Record_test	tc28	/Test Structure/Auto...
□	Record_test 1	tc29	/Test Structure/Auto...
□	Record_test 2	tc30	/Test Structure/Auto...

Figure 264 – New Scheduler

3.3.18.2 Delete Scheduler

For deleting a scheduler, press the "Delete" button. The button is enabled if at least one scheduler is selected.

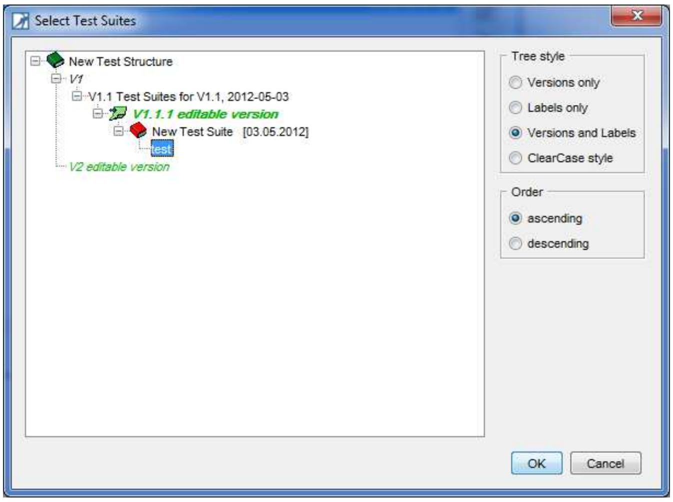


Name	Owner	Created	Updated	Updator	Host	Start until
test	Strasser Johanna	16.05.2012 11:29:37				31.05.2012 12:...

Figure 265 – Delete Scheduler

3.3.18.3 Import Scheduler

Instead of creating a scheduler from the scratch, importing schedulers from other test suites is possible. If you press the **Import**-button, a new window opens with the whole version tree. After selecting the scheduler which has to be imported, **OK** has to be pressed.



New Test Structure
V1
V1.1 Test Suites for V1.1, 2012-05-03
V1.1 editable version
New Test Suite [03.05.2012]
test
V2 editable version

Tree style
○ Versions only
○ Labels only
● Versions and Labels
○ ClearCase style

Order
● ascending
○ descending

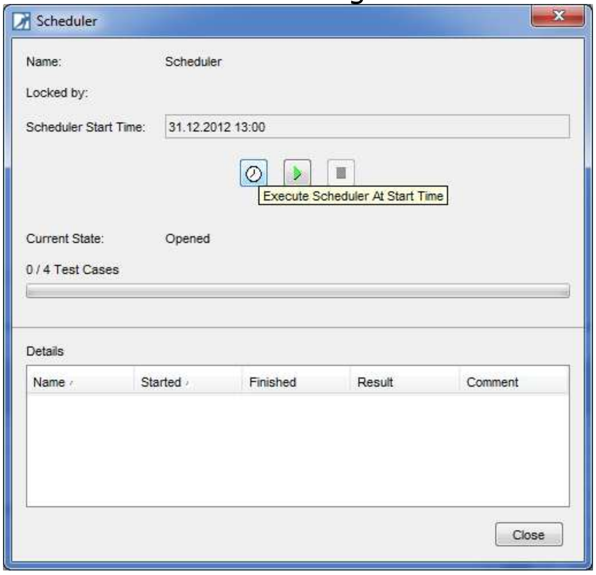
Figure 266 – Import Scheduler

After clicking OK, the scheduler is displayed in table at register **Scheduler**. Only existing test cases in current test suite can be applied to new scheduler.

3.3.18.4 Run Scheduler

For running a scheduler, press “Execute” – button. A new window opens where scheduler can be started in two different modes:

- Run at start time: The scheduler is locked and waits until start date. The scheduler starts working at defined start date.



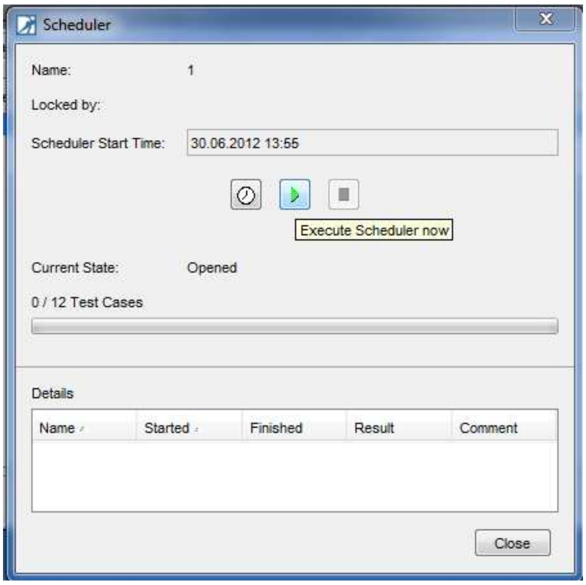
Name: Scheduler
Locked by:
Scheduler Start Time: 31.12.2012 13:00
Current State: Opened
0 / 4 Test Cases

Details

Name	Started	Finished	Result	Comment

Figure 267 – Execute Scheduler at Start Date

- Execute scheduler manually: The scheduler is locked and starts executing test cases immediately.



Scheduler

Name: 1

Locked by:

Scheduler Start Time: 30.06.2012 13:55

Execute Scheduler now

Current State: Opened

0 / 12 Test Cases

Details

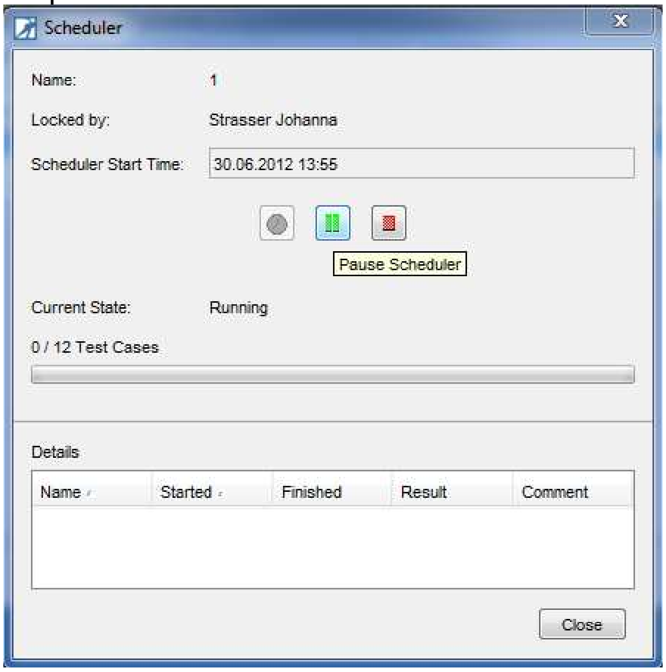
Name	Started	Finished	Result	Comment

Close

Figure 268 – Execute Scheduler

A running scheduler can be paused or stopped:

- **Pause:** When pressing "Pause", current test case execution is finished, the scheduler stops executing test cases and running mode is "pause". If the scheduler is paused, scheduler remains locked and user can continue or stop the scheduler.



Scheduler

Name: 1

Locked by: Strasser Johanna

Scheduler Start Time: 30.06.2012 13:55

Pause Scheduler

Current State: Running

0 / 12 Test Cases

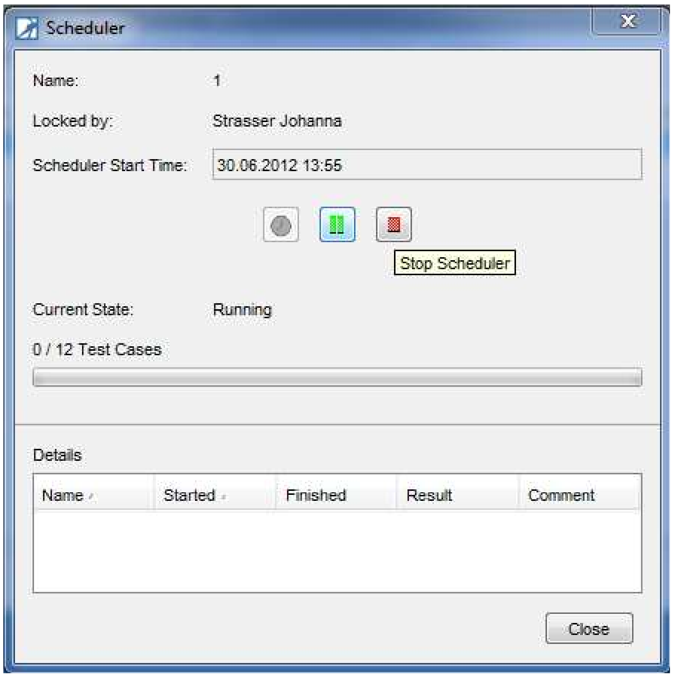
Details

Name	Started	Finished	Result	Comment

Close

Figure 269 – Pause Scheduler

- **Stop:** When pressing "Stop", current test case execution is finished and running mode is stopped. If a scheduler is stopped, the scheduler is unlocked and user cannot continue.



Name: 1

Locked by: Strasser Johanna

Scheduler Start Time: 30.06.2012 13:55

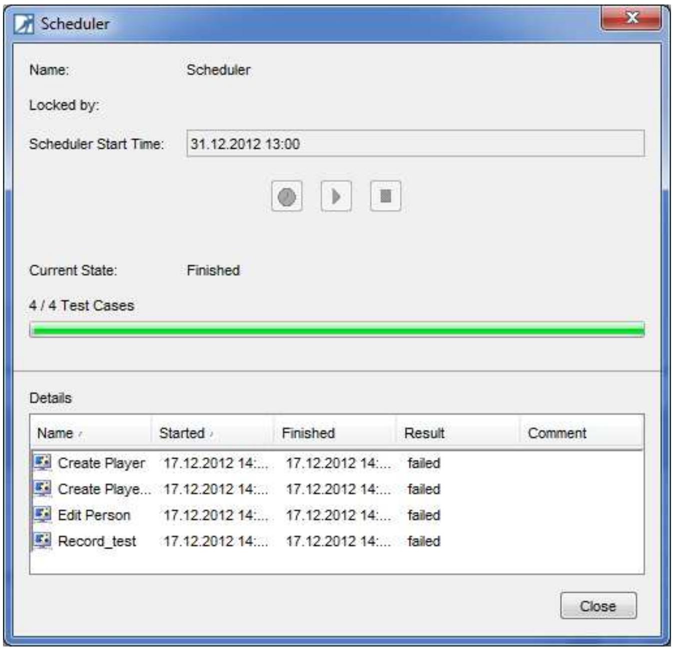
Current State: Running

0 / 12 Test Cases

Details

Name	Started	Finished	Result	Comment

Figure 270 – Stop Scheduler



Name: Scheduler

Locked by:

Scheduler Start Time: 31.12.2012 13:00

Current State: Finished

4 / 4 Test Cases

Details

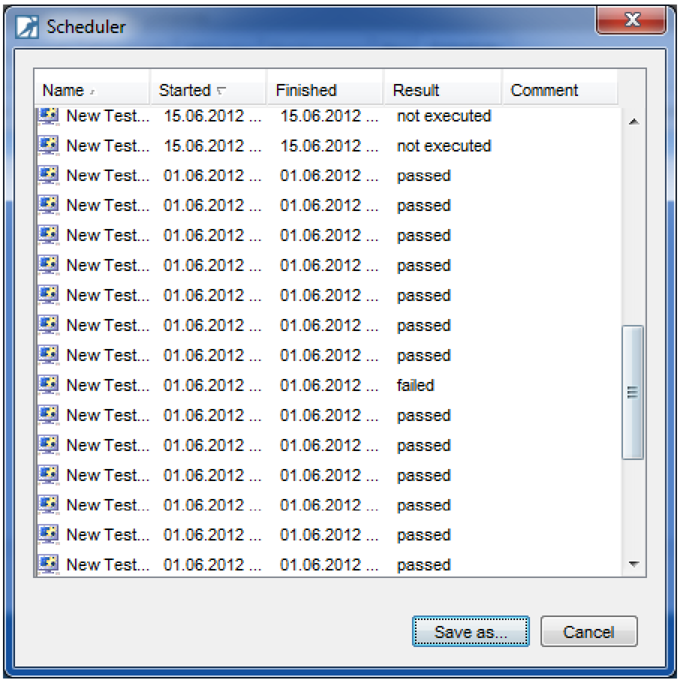
Name	Started	Finished	Result	Comment
Create Player	17.12.2012 14:...	17.12.2012 14:...	failed	
Create Playe...	17.12.2012 14:...	17.12.2012 14:...	failed	
Edit Person	17.12.2012 14:...	17.12.2012 14:...	failed	
Record_test	17.12.2012 14:...	17.12.2012 14:...	failed	

Figure 271 – Finished Scheduler

In **Details**, information about executed test cases are shown.

3.3.18.5 Protocol of Scheduler

All test case executions of a scheduler can be shown by pressing “Show Scheduler Protocol”.



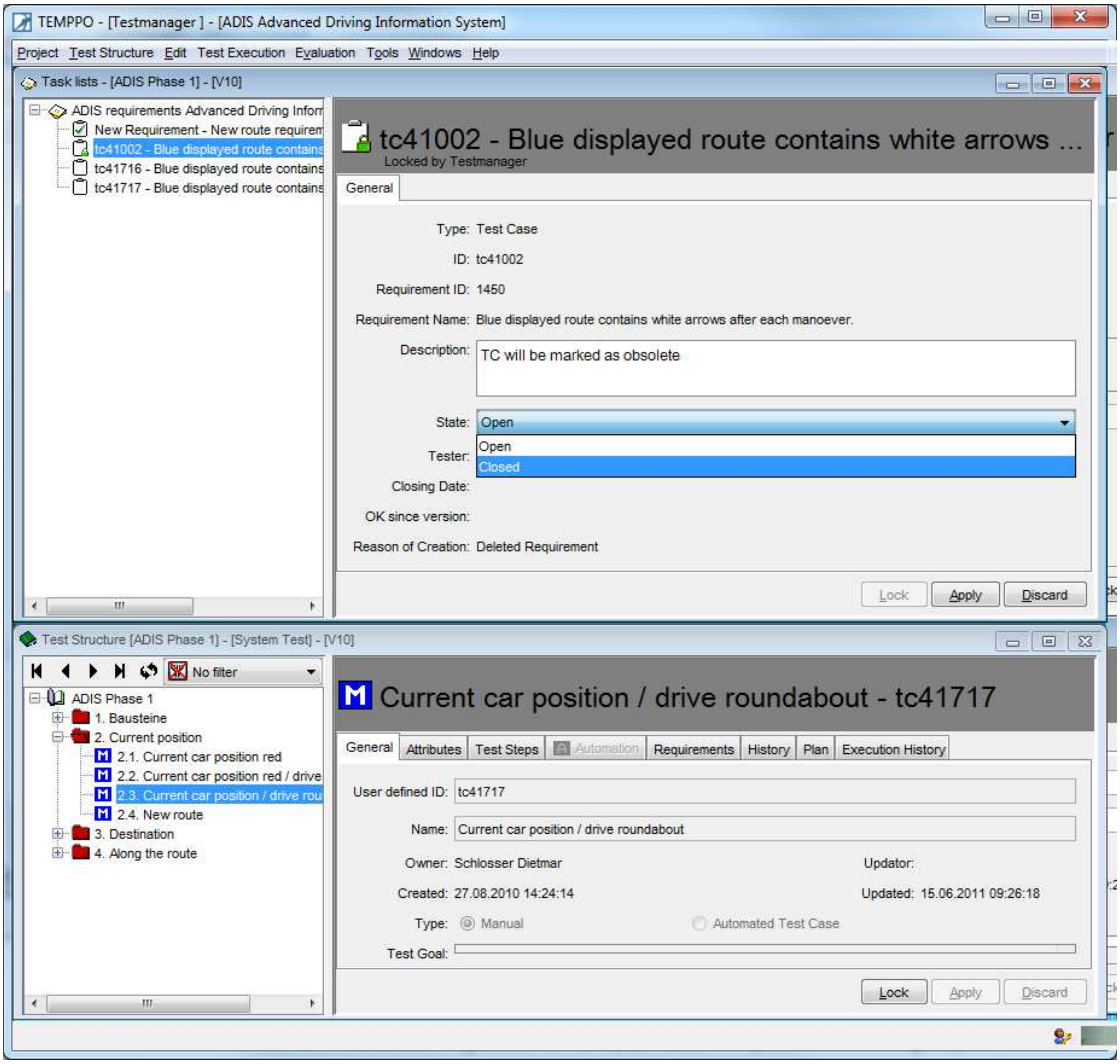
Name	Started	Finished	Result	Comment
New Test...	15.06.2012 ...	15.06.2012 ...	not executed	
New Test...	15.06.2012 ...	15.06.2012 ...	not executed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	failed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	
New Test...	01.06.2012 ...	01.06.2012 ...	passed	

Figure 272 – Scheduler Protocol

3.3.19Task lists

After **Test Structure > Apply requirement updates…**, a task list is generated. It can be viewed via menu **Test Structure > Task Lists**

All the **changed**, **new** and **deleted** requirements are listed. The lists are represented by a tree structure. Each list consists of one or many entries. All entries are marked as open the first time they are stored in the DB. If the user clicks an entry, the test case (or test package) of this entry is selected in the test structure window.



TEMPPO - [Testmanager] - [ADIS Advanced Driving Information System]

Task lists - [ADIS Phase 1] - [V10]

- ADIS requirements Advanced Driving Inforr
  - New Requirement - New route requirem
  - tc41002 - Blue displayed route contains
  - tc41716 - Blue displayed route contains
  - tc41717 - Blue displayed route contains

tc41002 - Blue displayed route contains white arrows ...

Locked by Testmanager

Type: Test Case

ID: tc41002

Requirement ID: 1450

Requirement Name: Blue displayed route contains white arrows after each manoever.

Description: TC will be marked as obsolete

State: Open

Tester:

Closing Date:

OK since version:

Reason of Creation: Deleted Requirement

Test Structure [ADIS Phase 1] - [System Test] - [V10]

- ADIS Phase 1
  - 1. Bausteine
  - 2. Current position
    - 2.1. Current car position red
    - 2.2. Current car position red / drive
    - 2.3. Current car position / drive rou
    - 2.4. New route
  - 3. Destination
  - 4. Along the route

Current car position / drive roundabout - tc41717

User defined ID: tc41717

Name: Current car position / drive roundabout

Owner: Schlosser Dietmar

Updator:

Created: 27.08.2010 14:24:14

Updated: 15.06.2011 09:26:18

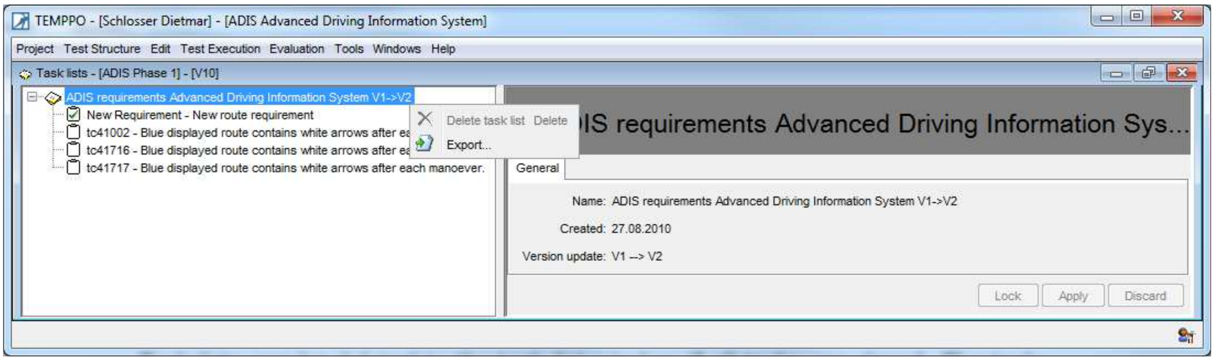
Type: ● Manual ○ Automated Test Case

Test Goal:

Figure 273 - Task due to deleted requirement

Task lists can be deleted in the task list window if all tasks are closed. The task list has to be selected and via the context menu item “Delete task list” a confirmation window comes up. Confirming with **OK** deletes the task list.

Task lists can be exported to Excel for further working on it.



ADIS requirements Advanced Driving Information System V1->V2
☑ New Requirement - New route requirement
□ tc41002 - Blue displayed route contains white arrows after ea
□ tc41716 - Blue displayed route contains white arrows after ea
□ tc41717 - Blue displayed route contains white arrows after each manoever.

IS requirements Advanced Driving Information Sys...

Name: ADIS requirements Advanced Driving Information System V1->V2
Created: 27.08.2010
Version update: V1 --> V2

Figure 274 – Task List

3.3.20Settings

TEMPPO differs between global and application specific settings.

3.3.20.1 Global settings

Global settings concern all TEMPPO applications – Test Manager, Requirement Manager, Administrator and Manager – i.e. when a global setting is changed, it's changed for all applications. If an application is running an a global setting is changed, the change will be visible after the next start of that application.

The global settings dialog can be opened via the menu **Global Settings** shortcut in the TEMPPO start menu.

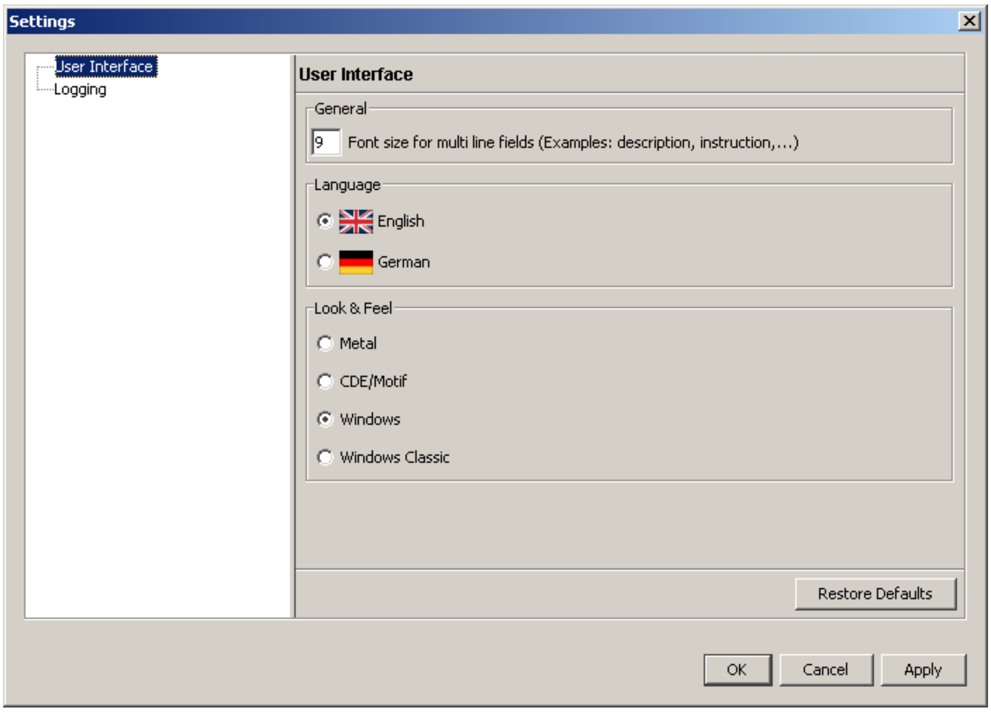
Following settings are global:

- User Interface
- Logging

3.3.20.1.1 User Interface

The user interface can be configure via following settings:

- Font size for multi-line fields
- Language
- Look & Feel



9 Font size for multi line fields (Examples: description, instruction,...)

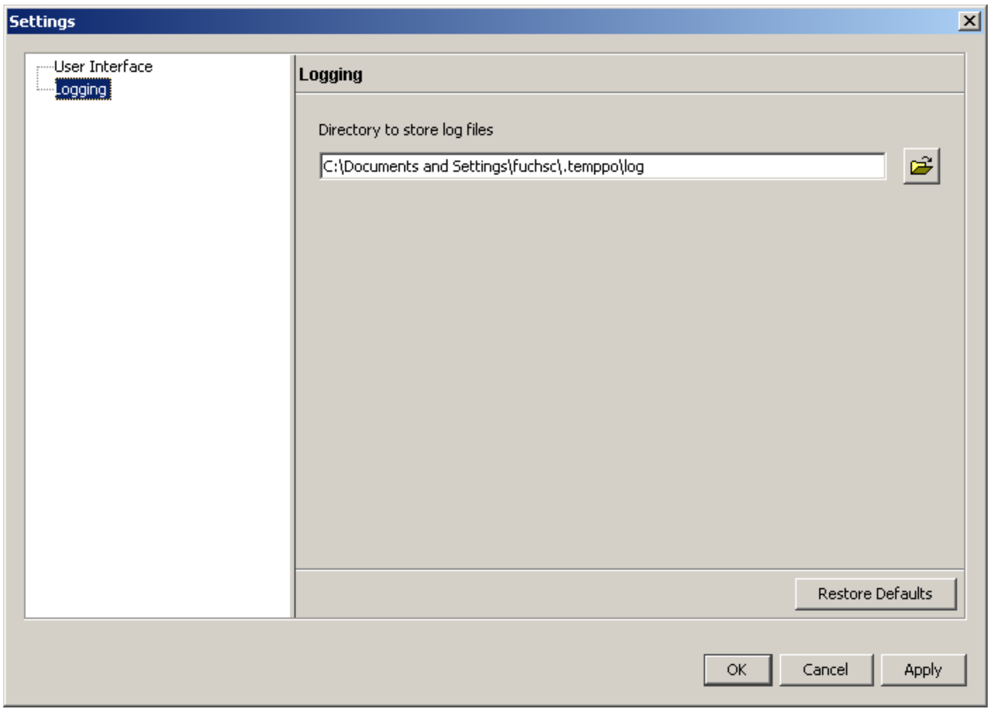
Language
● English
○ German

Look & Feel
○ Metal
○ CDE/Motif
● Windows
○ Windows Classic

Figure 275 – Global Settings: User Interface

3.3.20.1.2 Logging

You can also change the location of TEMPPO's log directory. Default value is the user's profile.

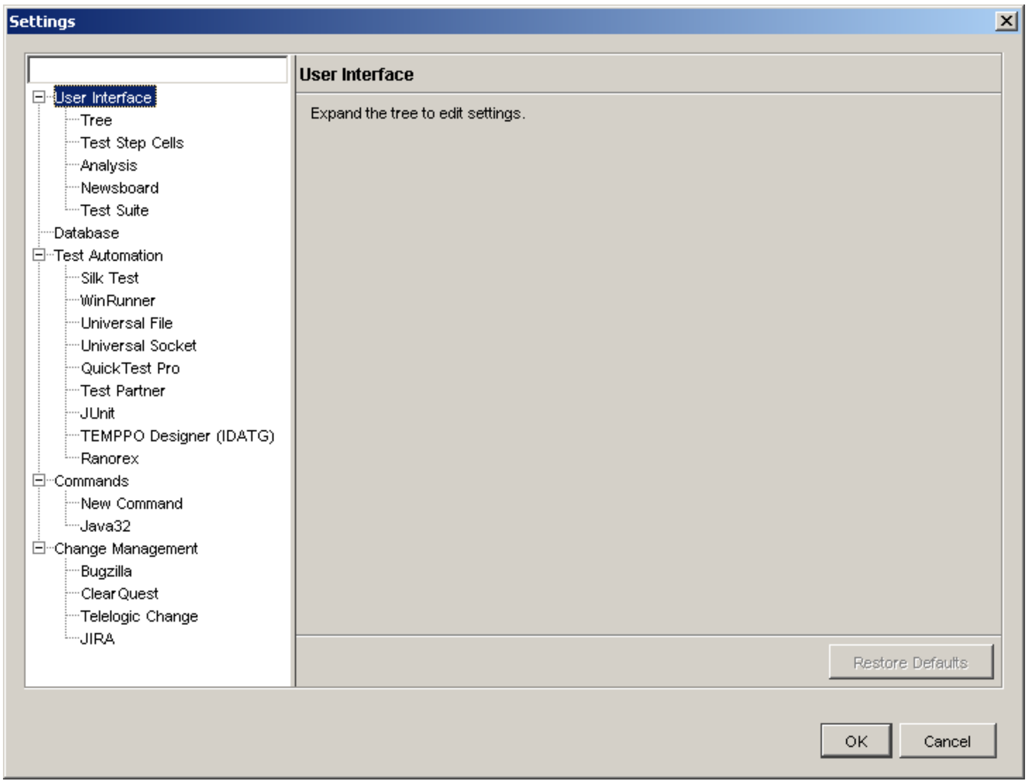


Directory to store log files
C:\Documents and Settings\fuchsc\.temppo\log

Figure 276 – Global Settings: Logging

3.3.20.2 TEMPPO Test Manager Settings

Specific settings e.g. tree display, automation tools, etc. can be configured in the settings dialog in TEMPPO. This dialog is opened via **Windows > Settings**. On the left side, there is a tree where the user can select the setting group which should be configured. On the right side all single settings can be edited (see Figure 277).



Expand the tree to edit settings.

Figure 277 – Settings

Via button **Restore Defaults** all settings of the selected setting group are restored to the default values.

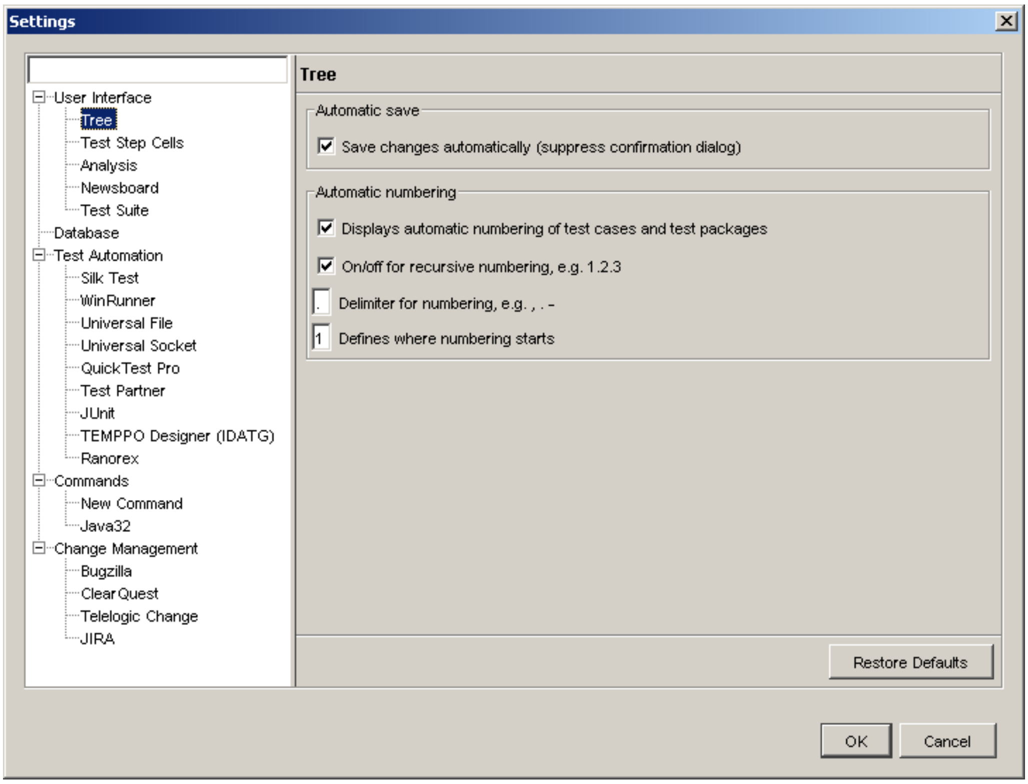
Following settings can be configured in TEMPPO:

- User Interface
  - Tree
  - Test Step Cells
  - Analysis
  - Newsboard
  - Test suite
- Database settings
- Test automation:
  - Silk Test
  - WinRunner
  - Universal File
  - Universal Socket
  - QuickTest Pro
  - Test Partner
  - JUnit
  - TEMPPO Designer (IDATG)
  - Ranorex
- Commands
- Change Management
  - Bugzilla
  - IBM Telelogic Change
  - IBM Rational Clearquest
  - JIRA

3.3.20.2.1 Tree

Tree settings which can be changed are, see Figure 278.

Property	Default value	Description
Display automatic numbering	deactivated	Switches on/off to display the internal order of items in the tree.
Recursive	activated	Switches on/off to show the order recursive, e.g. 10.7.03.
Delimiter	.	Specifies the delimiter for order numbers (e.g. delimiter in 10.7.3 is .).
Base	1	Specifies the base for order numbers, i.e. the number added to the internal order, which is 0 based.



Tree

Automatic save

☑ Save changes automatically (suppress confirmation dialog)

Automatic numbering

☑ Displays automatic numbering of test cases and test packages

☑ On/off for recursive numbering, e.g. 1.2.3

. Delimiter for numbering, e.g. , . -

1 Defines where numbering starts

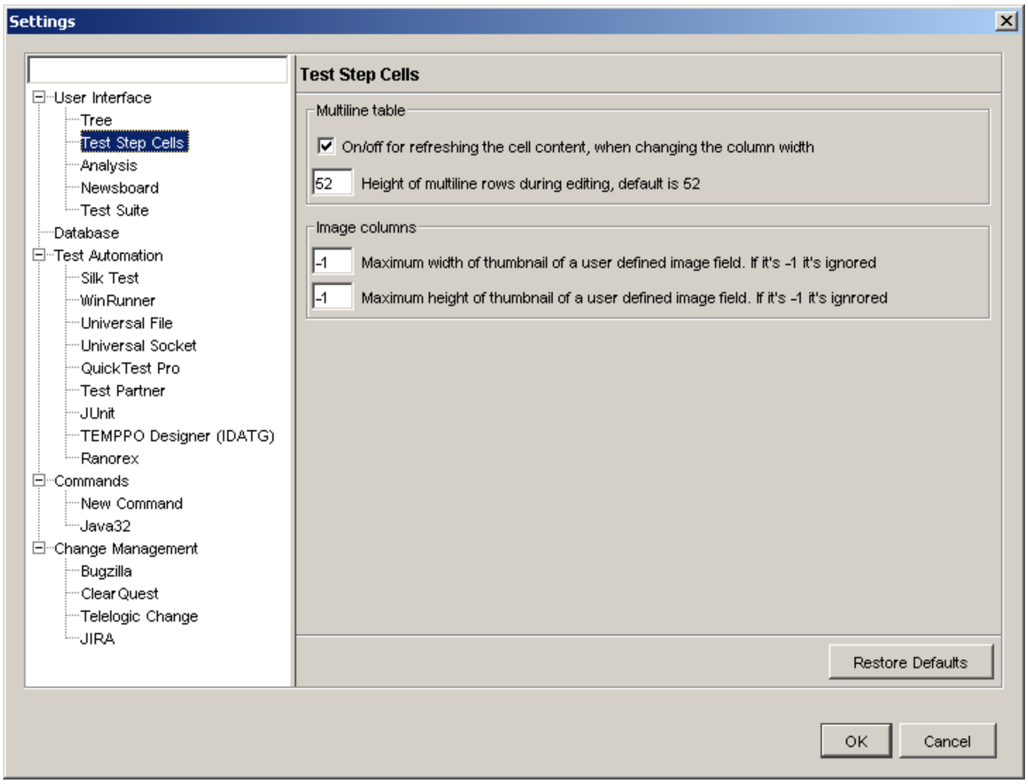
Figure 278 – Settings: Tree

3.3.20.2.2 Test Step Cells

For the test step cells some settings can be changed, see Figure 279.

Property	Default value	Description
Refresh when resizing	activated	Switches on/off refreshing the row heights in a multi line table when changing column widths or table bounds.
Multiline height	52	Sets the minimum height of the edited row (in pixels) in a multi line table.

		You cannot set the number of lines for the test step editor, but the pixels. 52 pixel correspond to 3 lines.
Image column – image width	-1	Maximum width of thumbnail of a user defined image field. If it’s -1 its ignored.
Image column – image height	100	Maximum height thumbnail of a user defined image field. If it’s -1 its ignored.



Test Step Cells

Multiline table

☑ On/off for refreshing the cell content, when changing the column width

52 Height of multiline rows during editing, default is 52

Image columns

-1 Maximum width of thumbnail of a user defined image field. If it's -1 it's ignored

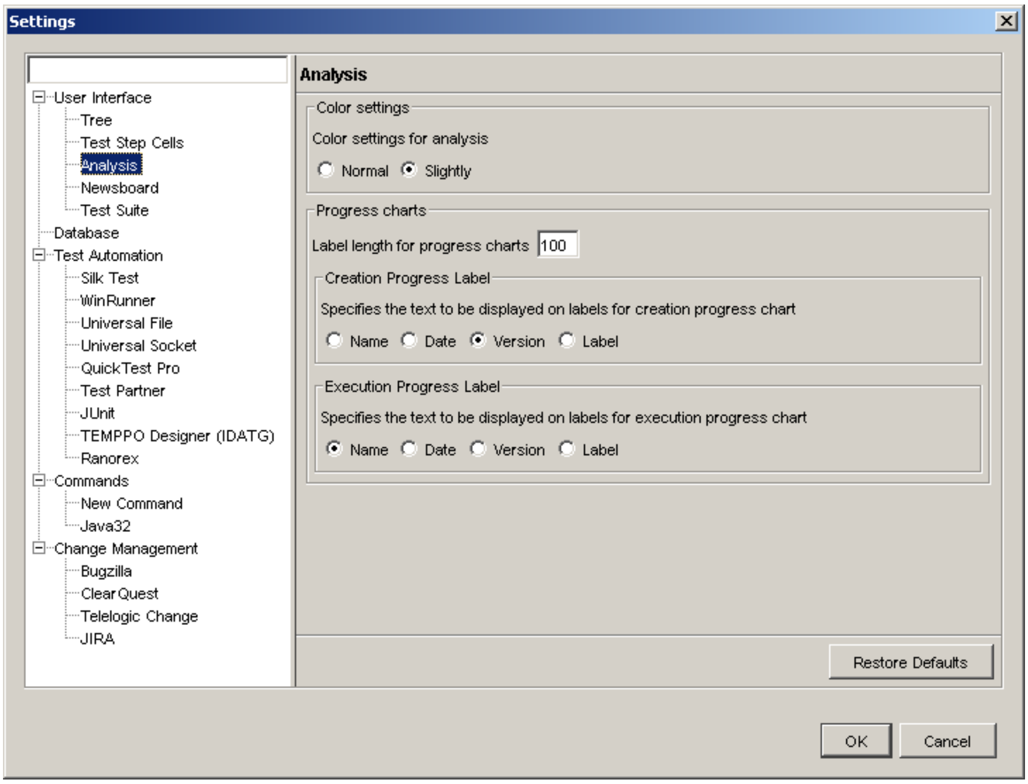
-1 Maximum height of thumbnail of a user defined image field. If it's -1 it's ignrored

Figure 279 – Settings: Test Step Cells

3.3.20.2.3 Analysis

You can change the settings for analysis:

Property	Default value	Description
Color settings	slightly	For charts, analysis you can select different colors
Label length for progress chart	100	Limits the label length of the requirement names in order to keep the chart “readable”
Creation Progress Label	Version	Specifies the text to be displayed on creation progress chart labels.
Execution Progress Label	Date	Specifies the text to be displayed on execution progress chart labels.



Analysis

Color settings

Color settings for analysis

○ Normal ● Slightly

Progress charts

Label length for progress charts 100

Creation Progress Label

Specifies the text to be displayed on labels for creation progress chart

○ Name ○ Date ● Version ○ Label

Execution Progress Label

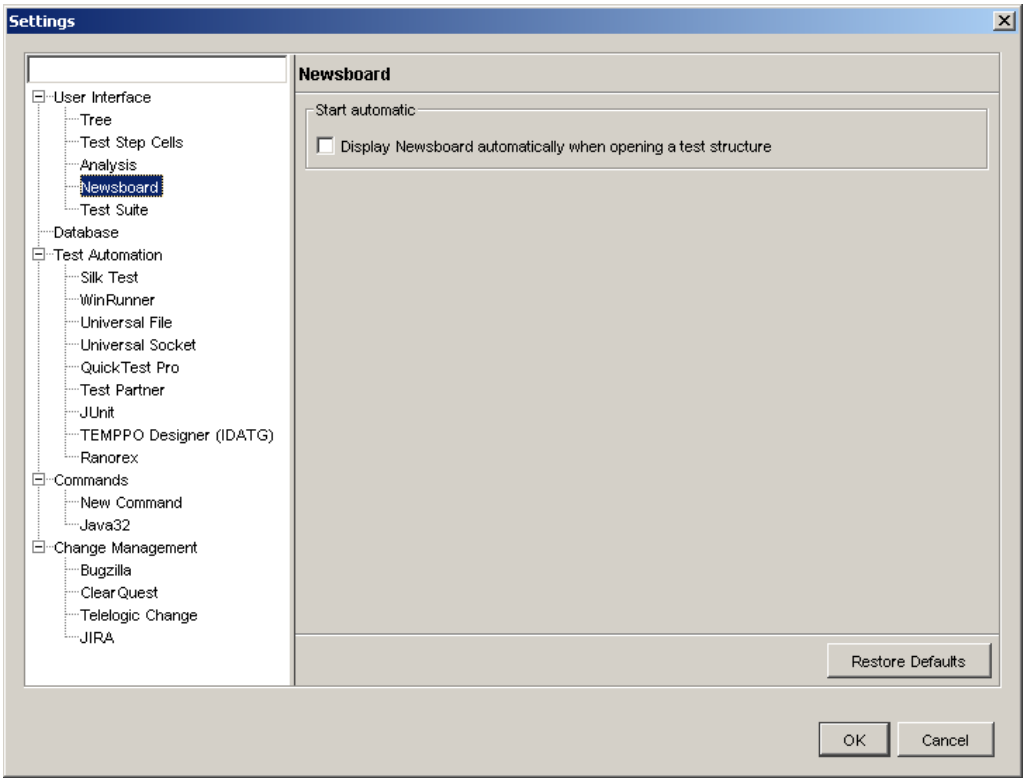
Specifies the text to be displayed on labels for execution progress chart

● Name ○ Date ○ Version ○ Label

Figure 280 – Settings: Analysis

3.3.20.2.4 Newsboard

You can configure, if Newsboard window is automatically opened, if a test structure is opened.



Newsboard

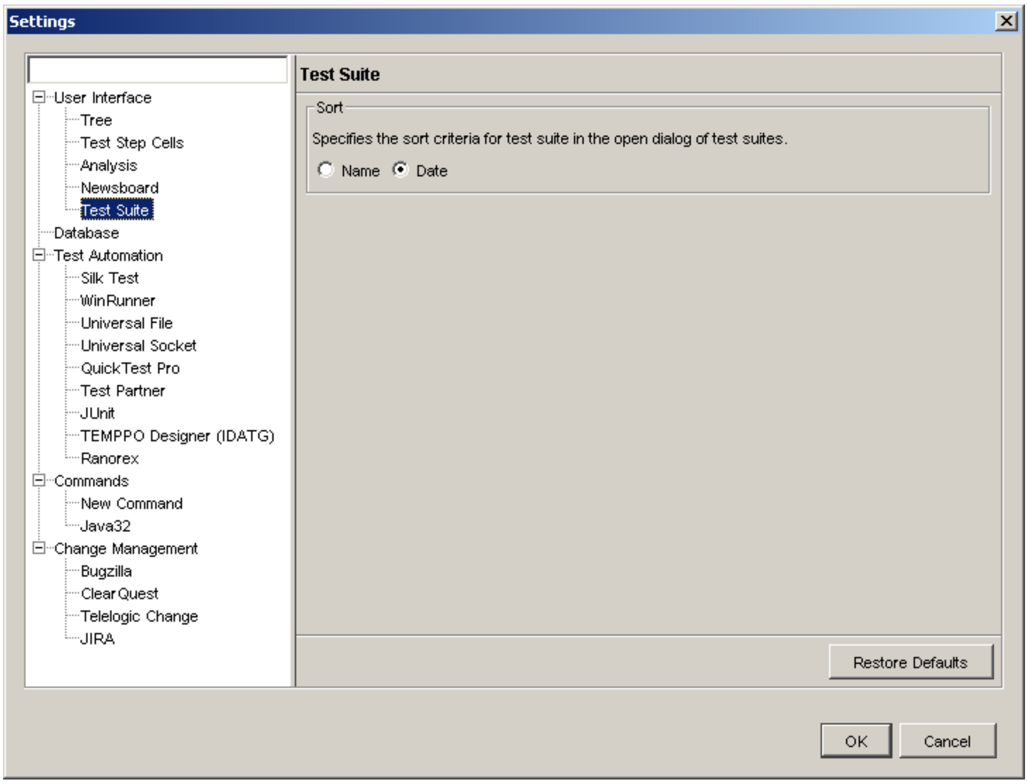
Start automatic

☐ Display Newsboard automatically when opening a test structure

Figure 281 – Settings: Newsboard

3.3.20.2.5 Test suite

You specify, if test suites are sorted by name or date in the open dialog.



Test Suite

Sort

Specifies the sort criteria for test suite in the open dialog of test suites.

○ Name ● Date

Figure 282 - Settings: Test suite

3.3.20.2.6 Database

For the settings of database see Figure 283.

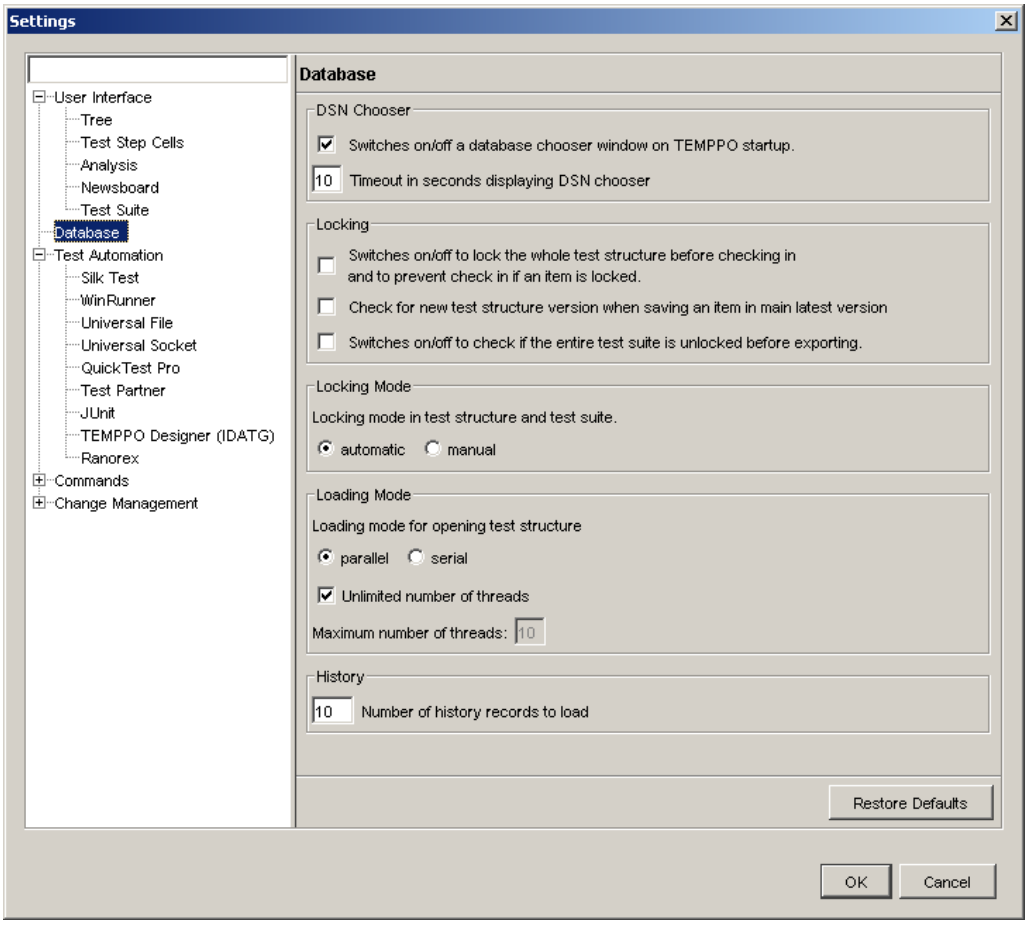
Property	**Default value**	**Description**
Show DSN Chooser on Startup	deactivated	Switches on/off a DB chooser on TEMPPO startup.
DSN Chooser Timeout (seconds)	5	Sets the timeout (in seconds) for committing the chooser automatically.
Switches on/off to lock…	deactivated	Lock whole test structure before checking-in in case of test suite creation.
Check new test structure version	deactivated	Check for new test structure version when saving an item in latest version This setting is used together with deactivated previous setting. Example: User1: edits a TC in latest version User2: creates new test suite User1: saves TC and the new test structure version is reloaded.
Check Locks in Test Suite before Exporting	activated	Switches on/off check of locks in test suite before exporting it.
Locking mode	automatic	Sets the locking mode
Loading mode	Parallel (unlimited threads)	When opening or refreshing a test structure there is a possibility to speed up loading data. If you want to do that, you have to select **Parallel.** **Parallel:** If you work on a single client PC it is recommended to select the checkbox **Unlimited number of threads**. For loading time CPU usage goes up. If you work on a terminal server it is recommended to specify a number of threads. Please ask SiTEMPPO team to specify the best value for your test environment. **Serial**: Data is loaded serialized. Only recommended when a small amount of data.
History entries	10	Number of history records loaded at once from DB. Further records can be loaded on demand.

Recommendations:

For parallel mode, the number of threads should be `2*number of processors` on database server.

The setting for closing unused database connections depends on usage of TEMPPO:

- If you are opening frequently different test structure or versions, then you should define a longer duration. Then opened database connections are reused and no additional resources are needed permanently for connecting/disconnecting to database.
  For example: 30 minutes
- If you are working only in one test structure version for a long time, then you can define a short duration. Opened database connections are closed and used resources are released.
  For example: 10 minutes



Database

DSN Chooser

☑ Switches on/off a database chooser window on TEMPPO startup.

10 Timeout in seconds displaying DSN chooser

Locking

□ Switches on/off to lock the whole test structure before checking in and to prevent check in if an item is locked.

□ Check for new test structure version when saving an item in main latest version

□ Switches on/off to check if the entire test suite is unlocked before exporting.

Locking Mode

Locking mode in test structure and test suite.

● automatic ○ manual

Loading Mode

Loading mode for opening test structure

● parallel ○ serial

☑ Unlimited number of threads

Maximum number of threads: 10

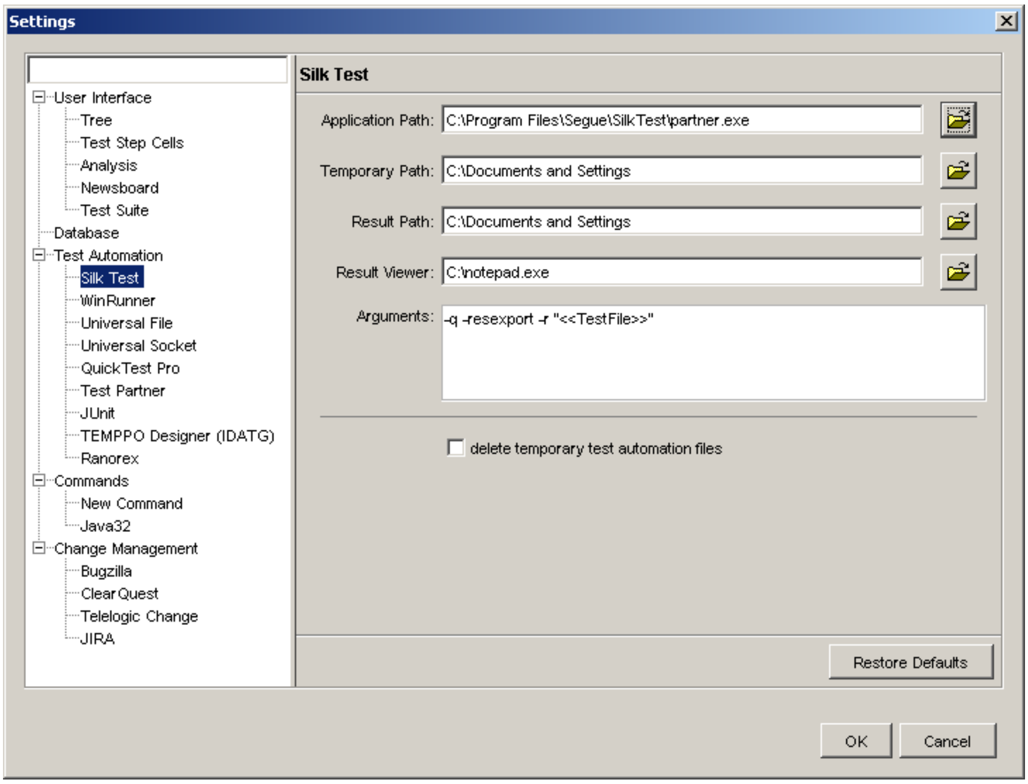
History

10 Number of history records to load

Figure 283 – Settings: Database

3.3.20.2.7 Test Automation

3.3.20.2.7.1 WinRunner and SilkTest



Silk Test

Application Path: C:\Program Files\Segue\SilkTest\partner.exe

Temporary Path: C:\Documents and Settings

Result Path: C:\Documents and Settings

Result Viewer: C:\notepad.exe

Arguments: -q -resexport -r "<<TestFile>>"

□ delete temporary test automation files

Figure 284: Test Automation (SilkTest, WinRunner)

If you plan to manage automated test cases in TEMPPO, you define the automation settings here. First of all you have to choose the automation tool, which will run your test cases. For WinRunner, SilkTest the meaning of the settings is as follows:

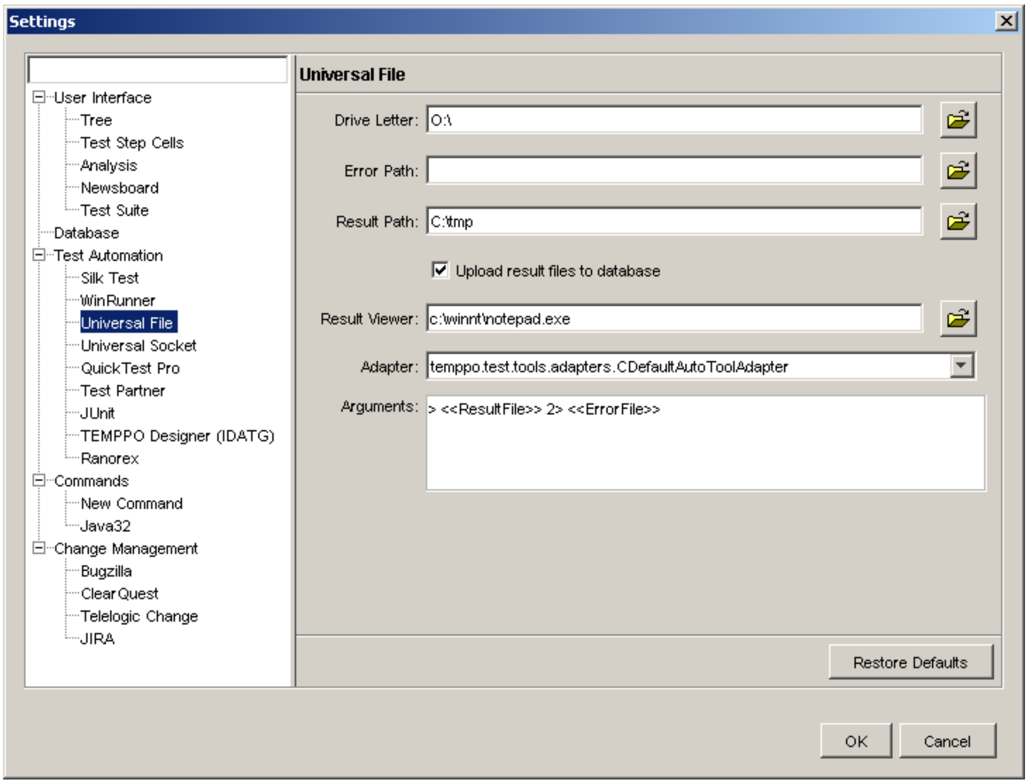
- **Application Path**: Executable of the automation tool.
- **Temporary Path**: Directory for temporary automation files created by TEMPPO during the execution of automated test cases.
- **Result Path**: Directory for result files created by the automation tool. The created result files are read from TEMPPO for obtaining the automation results.
- **Result Viewer**: The result viewer is for opening and viewing the result files. It depends on the result file format, which viewer can be taken (use a text editor e.g. notepad for WinRunner result files, use QuickTest Professional Result Viewer for QuickTest, use SilkTest for SilkTest result files).
- **Arguments**: The automation tool is called via its command line interface. This field shows the arguments which are passed to the automation tool during the call. Normally the default arguments should fulfill your needs for test automation, but for special cases the arguments can be adapted (study the command line specification of the automation tool first!). For the definition of the argument you can use the parameters <<TestFile>> (automation file to execute), <<ResultBaseDir>> (= Result Path) and <<ResultDir>> (generated directory for results during automation). Attention: Not every tool supports all parameters!

Use the Default button to restore the default arguments

If the check box in the bottom of the tab is checked, the temporary automation files will be deleted after automation.

3.3.20.2.7.2 Universal File Automation Interface

For UFAI (Universal File Automation Interface) you have to activate the automation tab and select **Universal File**.



Universal File

Drive Letter: O:\

Error Path:

Result Path: C:\tmp

☑ Upload result files to database

Result Viewer: c:\winnt\notepad.exe

Adapter: temppo.test.tools.adapters.CDefaultAutoToolAdapter

Arguments: > <<ResultFile>> 2> <<ErrorFile>>

Figure 285 – Settings: Universal File

The meanings of the parameters are as follows:

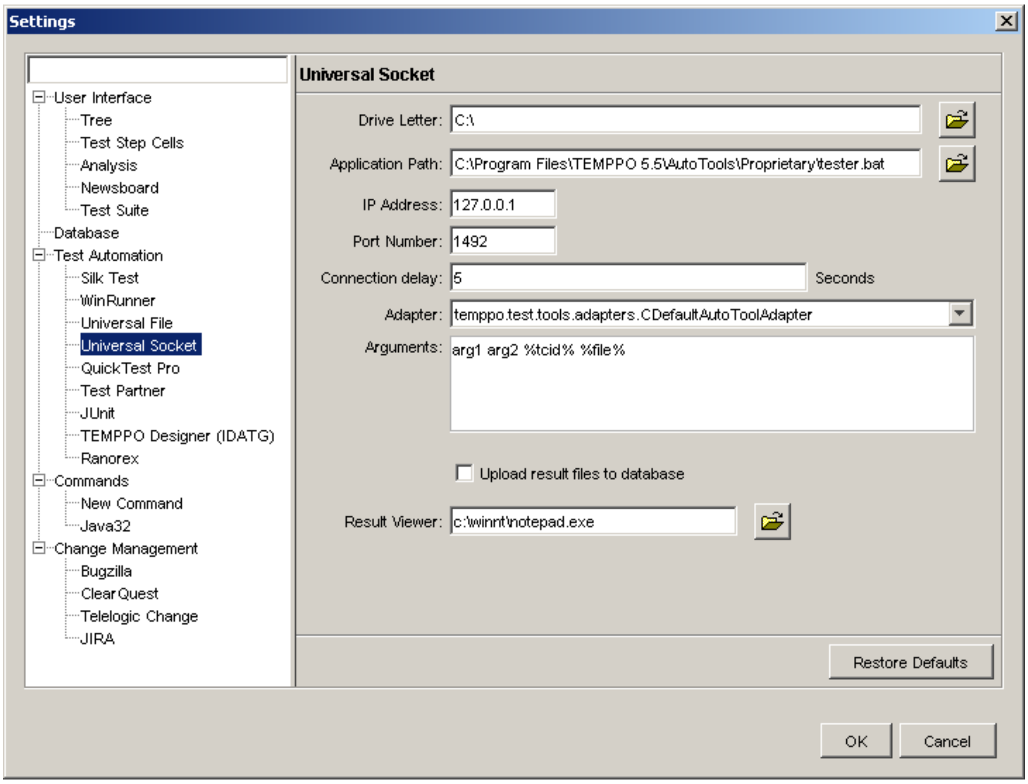
- **Drive Letter** is that letter, where path of all test executables start. For each test case a path has to be added without a letter drive. In this way it is possible to support scenarios, where test executables have to be started from varying drive letters. You only have to change the Drive Letter setting in the Administrator.
- **Result Path** (**mandatory)**: This is the directory, where all the result files are stored.
- **Upload result files to DB:** if checked, the result files will be loaded into the db; if unchecked the result file are stored in the file system.
- **Error Path: (mandatory):** The directory where all error output files are stored.
- **Result Viewer (optional):** An editor which is best suited for viewing result and output files.
- **Arguments (optional):** Command line arguments which are valid for all test executables. Contrarily individual arguments can be specified separately for each test case. Note, that the test executable is called with both argument strings concatenated.

3.3.20.2.7.3 Universal Socket Automation Interface

This interface allows test automation via a socket connection. I.e. when starting a test execution, TEMPPO will establish a local socket connection on a specified port, send the path to the script to execute over this socket and read the test results from this socket.

Via an adapter implementation it’s possible to handle the tool specific format of the delivered result output.

For configuration of USAI (Universal Socket Automation Interface) you have to activate the automation tab and select **Universal Socket**.



Universal Socket

Drive Letter: C:\

Application Path: C:\Program Files\TEMPPO 5.5\AutoTools\Proprietary\tester.bat

IP Address: 127.0.0.1

Port Number: 1492

Connection delay: 5 Seconds

Adapter: temppo.test.tools.adapters.CDefaultAutoToolAdapter

Arguments: arg1 arg2 %tcid% %file%

☐ Upload result files to database

Result Viewer: c:\winnt\notepad.exe

Figure 286 – Settings: Universal Socket

The meanings of the parameters are as follows:

- **Drive Letter** is that letter, where path of all test executables start. For each test case a path has to be added without a letter drive. In this way it is possible to support scenarios, where test executables have to be started from varying drive letters. You only have to change the Drive Letter setting in the Administrator.
- **Application path** must point to the executable, which will be launched by TEMPPO, if a socket connection is not possible when test execution was started. After starting the application, TEMPPO will wait a defined delay (to give the application enough time to startup), before retrying to open the socket connection. This delay is 5 seconds per default.
- The **IP address** from the host where the application is running has to be specified. If the host isn’t the local host (127.0.0.1) the application path cannot be configured. The reason for this limitation is that it is not possible to launch an application on a remote computer without any additional software.
- At **port number** the socket connection for automation will be opened.
- **Adapter** is the tool specific implementation for the handling of results received via the socket, to convert it into a format understandable by TEMPPO. Select here the implementation for your tool. For more information on adapter implementations see 13.7.

- **Arguments** are the command line arguments for the automation application.

In the arguments, parameters can be used for the configuration:

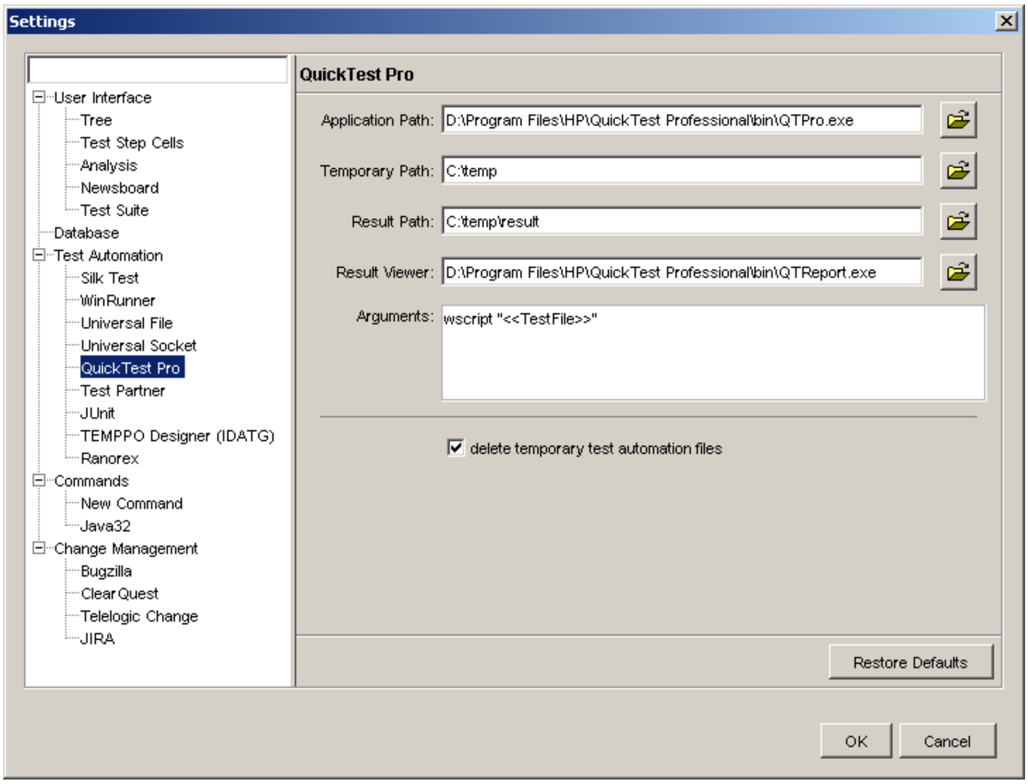
Parameter	Purpose
%dbtcid%	internal unique database id for the test case
%dbtsid%	internal unique database id for the test suite
%tcid%	user defined id of test case
%tca%	test case name
%tst%	test structure name
%tsu%	test suite name
%prj%	project name
%ver%	Version
%file%	automation file

–
– Example:
– Administrator configuration (Arguments):
– -tc %dbtcid% -ts %dbtsid% -f "%file%" –args
–
– Test case configuration (Additional Arguments):
– atw108s2;3;xyz
–
– Data sent over socket:
– -tc 213 –ts 1158 –f "x:\test scripts\jlms_checkout.xml" –args "atw108s2;3;xyz"

3.3.20.2.7.4 QuickTest Pro

For QuickTest Pro you have to specify the following parameters:

- **Application Path**: path to QTPro.exe
- **Temporary Path**: path where temporary files are stored
- **Result Path**: path where all the result runs are stored
- **Result viewer**: path to QTReport.exe



QuickTest Pro

Application Path: D:\Program Files\HP\QuickTest Professional\bin\QTPro.exe

Temporary Path: C:\temp

Result Path: C:\temp\result

Result Viewer: D:\Program Files\HP\QuickTest Professional\bin\QTReport.exe

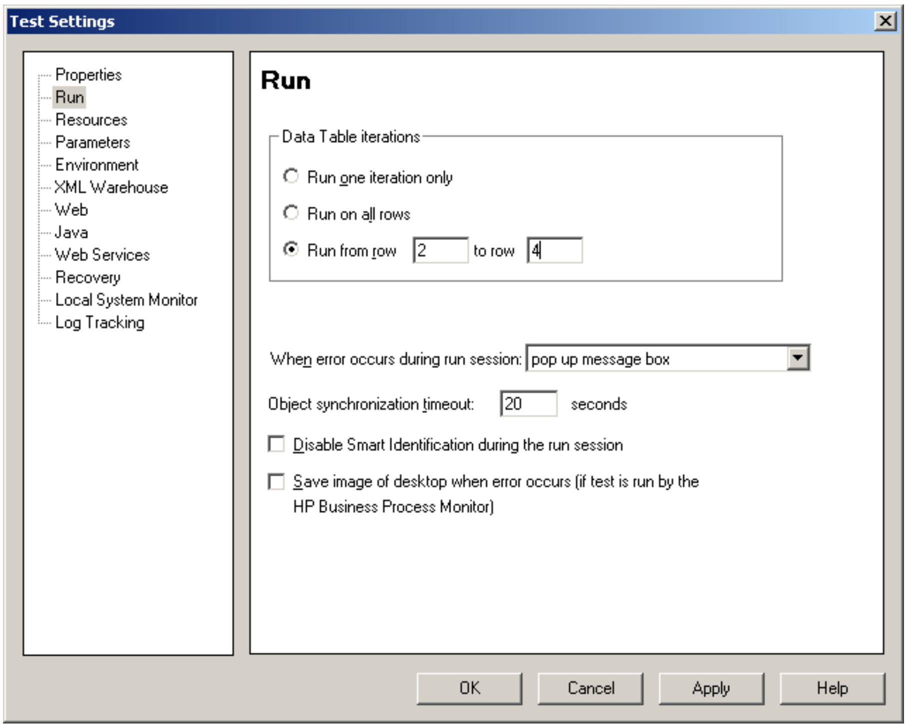
Arguments: wscript "<<TestFile>>"

☑ delete temporary test automation files

Figure 287 – Settings: QuickTest Pro

– TEMPPO uses the data table iterations from the QTP test case:

–



Run

Data Table iterations

○ Run one iteration only

○ Run on all rows

● Run from row 2 to row 4

When error occurs during run session: pop up message box

Object synchronization timeout: 20 seconds

□ Disable Smart Identification during the run session

□ Save image of desktop when error occurs (if test is run by the HP Business Process Monitor)

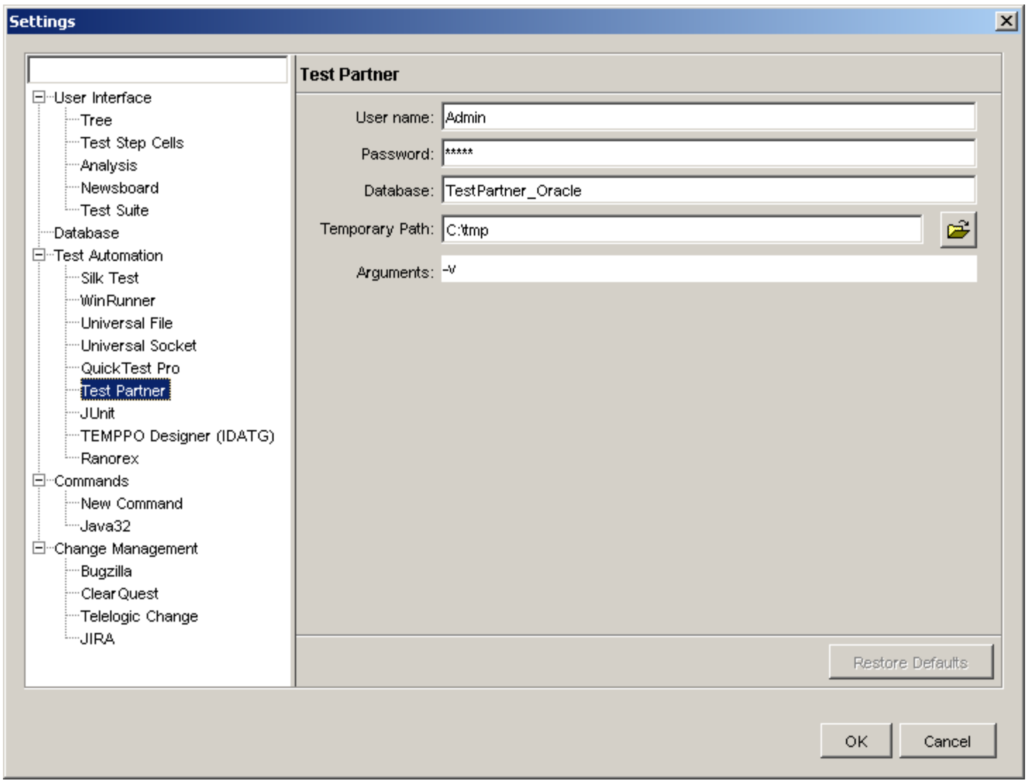
–

Figure 288 – QTP Run Settings

3.3.20.2.7.5 TestPartner

ⓘ **TestPartner requires .NET Framework 2.0**

For Compuware TestPartner you have to activate the automation tab and select **TestPartner**.



Test Partner

User name: Admin

Password: *****

Database: TestPartner_Oracle

Temporary Path: C:\tmp

Arguments: -v

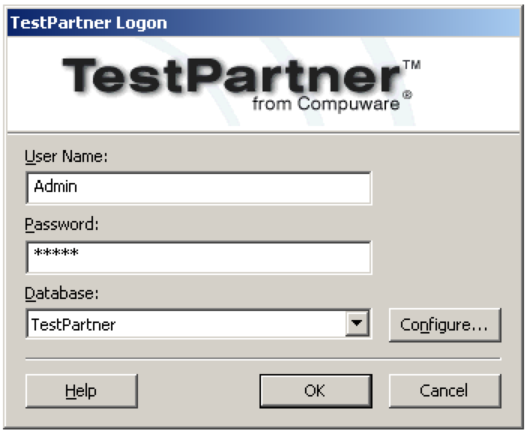
Figure 289 – Settings: TestPartner

The meanings of the parameters are as follows:

- **User Name**: User name in TestPartner for executing the automated test cases.
- **Password**: The password for the user in TestPartner.
- **Database**: Name of database when logging in to TestPartner (see Figure 289)
- **Temporary Path**: Has to be set for storing TestPatner's temporary data.

Use the Default button to restore the default arguments.

If the check box in the bottom of the tab is checked, the temporary automation files will be deleted after automation.



User Name: Admin

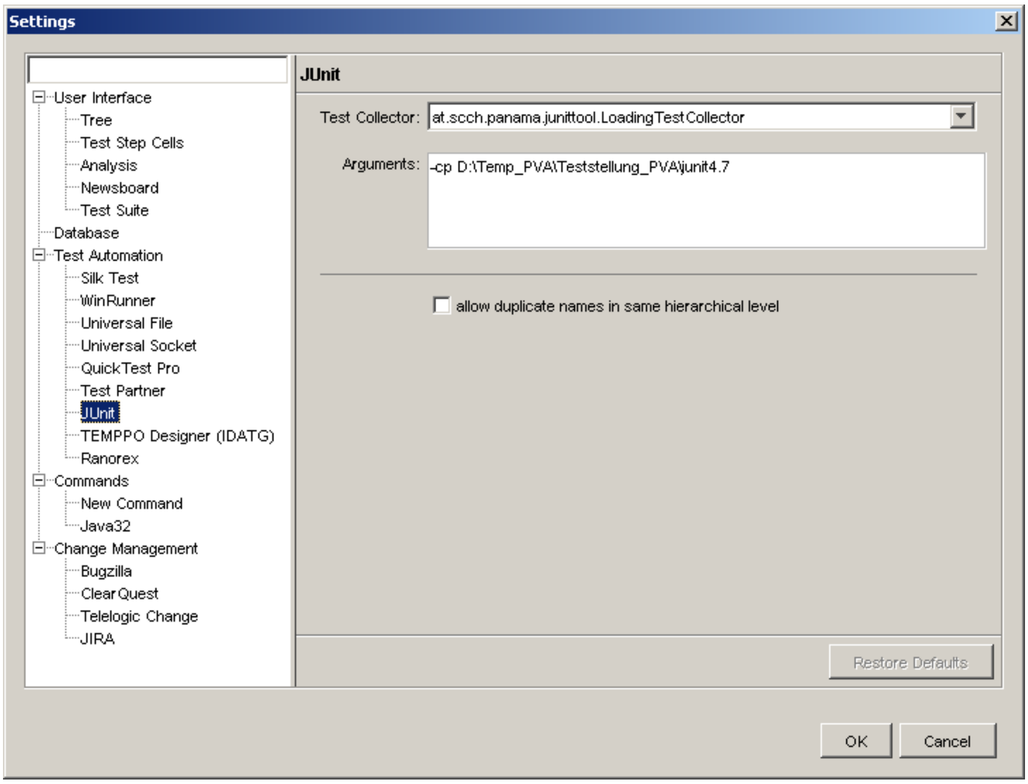
Password: *****

Database: TestPartner

Figure 290 – TestPartner Login

3.3.20.2.8 JUnit

For Junit see chapter 6.



JUnit

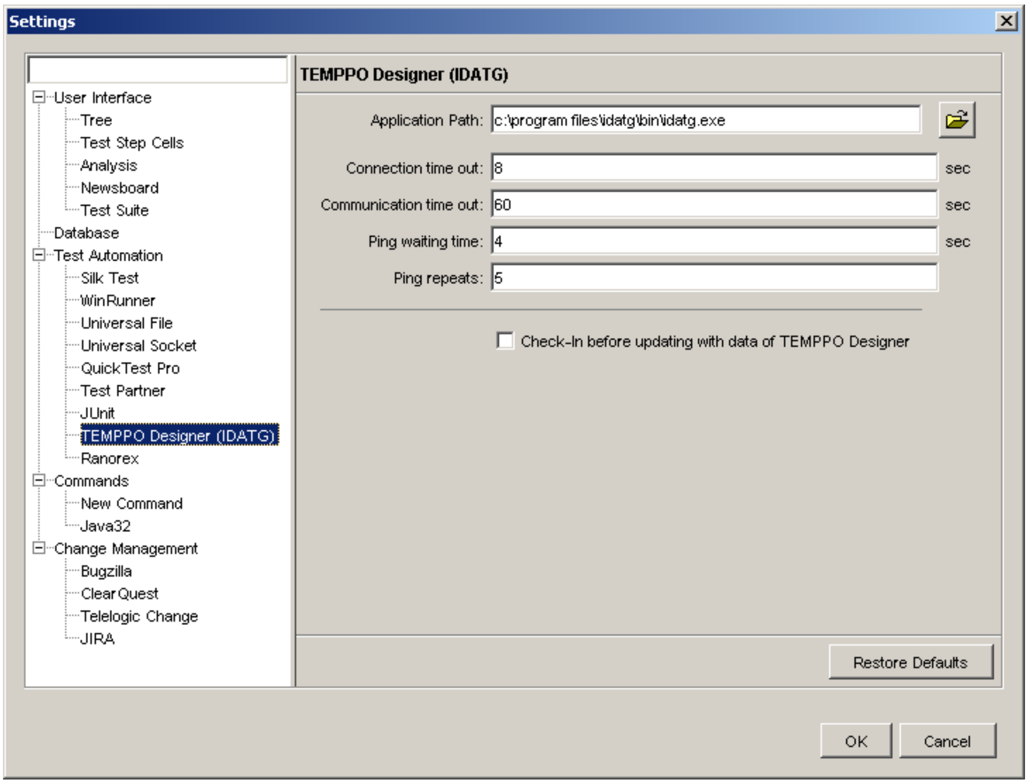
Test Collector: at.scch.panama.junittool.LoadingTestCollector

Arguments: -cp D:\Temp_PVA\Teststellung_PVA\junit4.7

☐ allow duplicate names in same hierarchical level

Figure 291 – Junit configuration

3.3.20.2.9 TEMPPO Designer (IDATG)



TEMPPO Designer (IDATG)

Application Path: c:\program files\idatg\bin\idatg.exe

Connection time out: 8 sec

Communication time out: 60 sec

Ping waiting time: 4 sec

Ping repeats: 5

☐ Check-In before updating with data of TEMPPO Designer

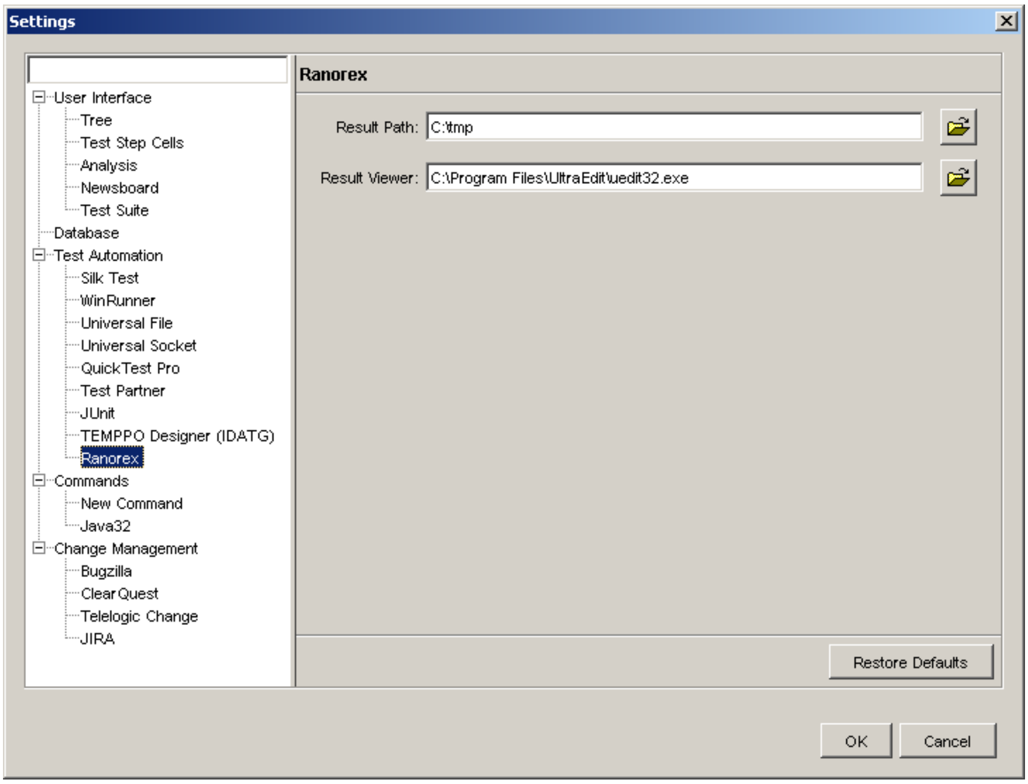
Figure 292 – Settings: TEMPPO Designer (IDATG)

- **Application path**: Path to IDATG.exe
- **Connection time out**: Time for connecting successfully with TEMPPO Designer (IDATG)

- **Communication time out:** Time out for an existing connection without any activities. After that time out TEMPPO sends a ping / heart beat and waits the Ping waiting time
- **Ping waiting time**: Maximal time to wait for an answer on a ping / heart beat.
- **Ping repeats**:Number of pings / heart beats sent to TEMPPO Designer (IDATG)

3.3.20.2.10 Ranorex

- **Result path**: Path to directory of the result file(s)
- **Result viewer**: Application to open result file(s)



Result Path: C:\tmp

Result Viewer: C:\Program Files\UltraEdit\uedit32.exe

Figure 293 – Settings: Ranorex

3.3.20.2.11 Commands

Commands are used for hyperlinks with parameters.

By pressing the **New Command** on the right side you can create a new command (see Figure 294). Then a new tree node is created with the new command (see Figure 295). A command is defined by its name, path and arguments. The name has to be unique.

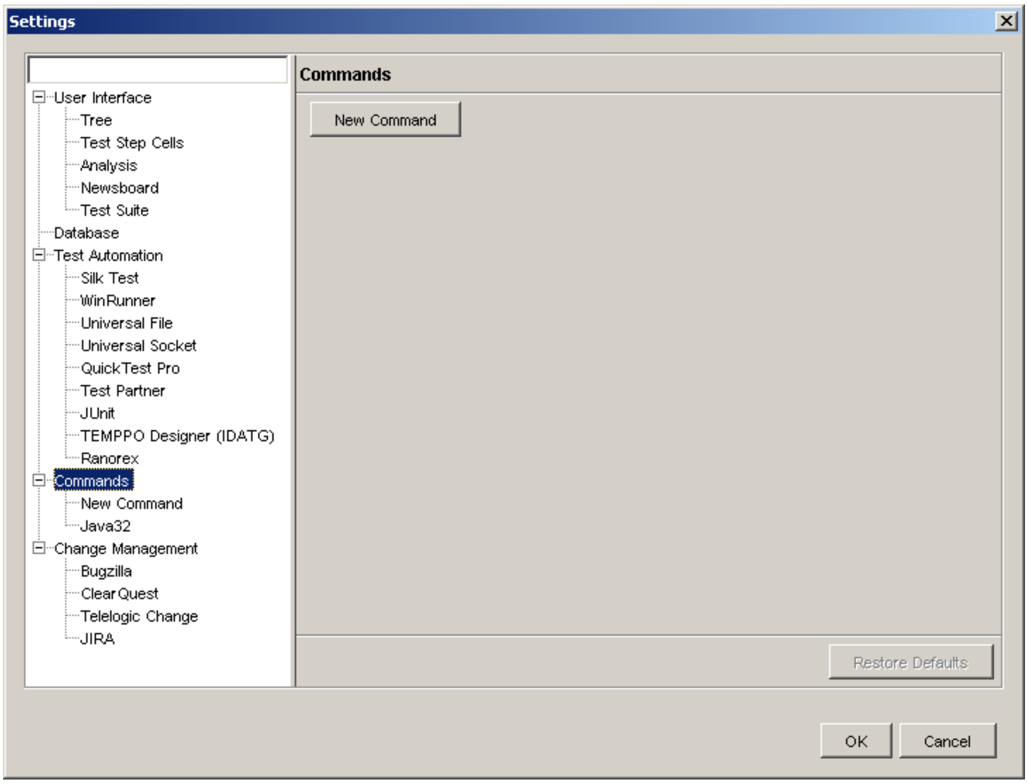
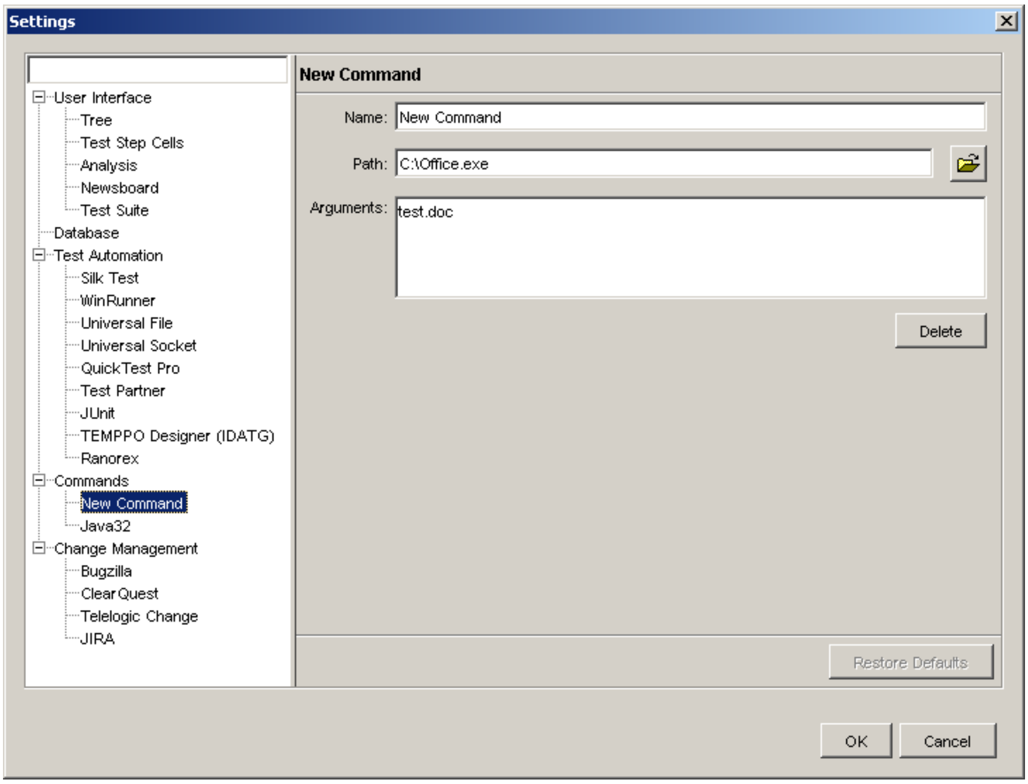


Figure 294 – Settings: Command



Name: New Command

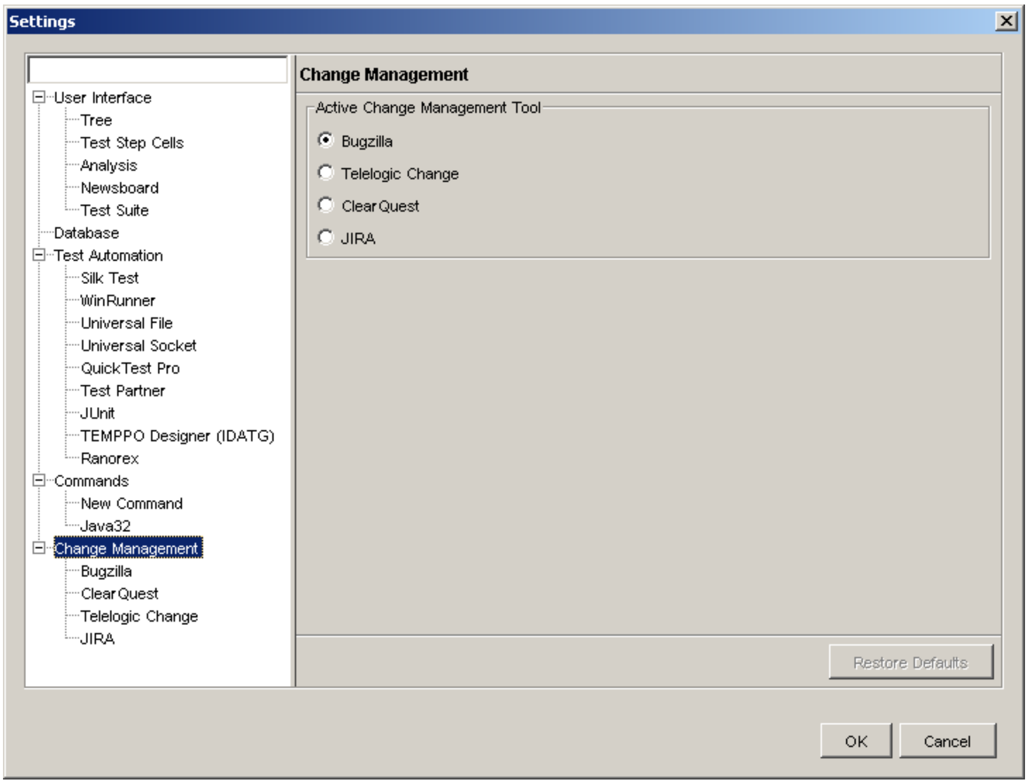
Path: C:\Office.exe

Arguments: test.doc

Figure 295 – New Command

3.3.20.2.12 Change Management

Select the change management tool which is used in TEMPPO for adding bugs to test steps.



Change Management

Active Change Management Tool

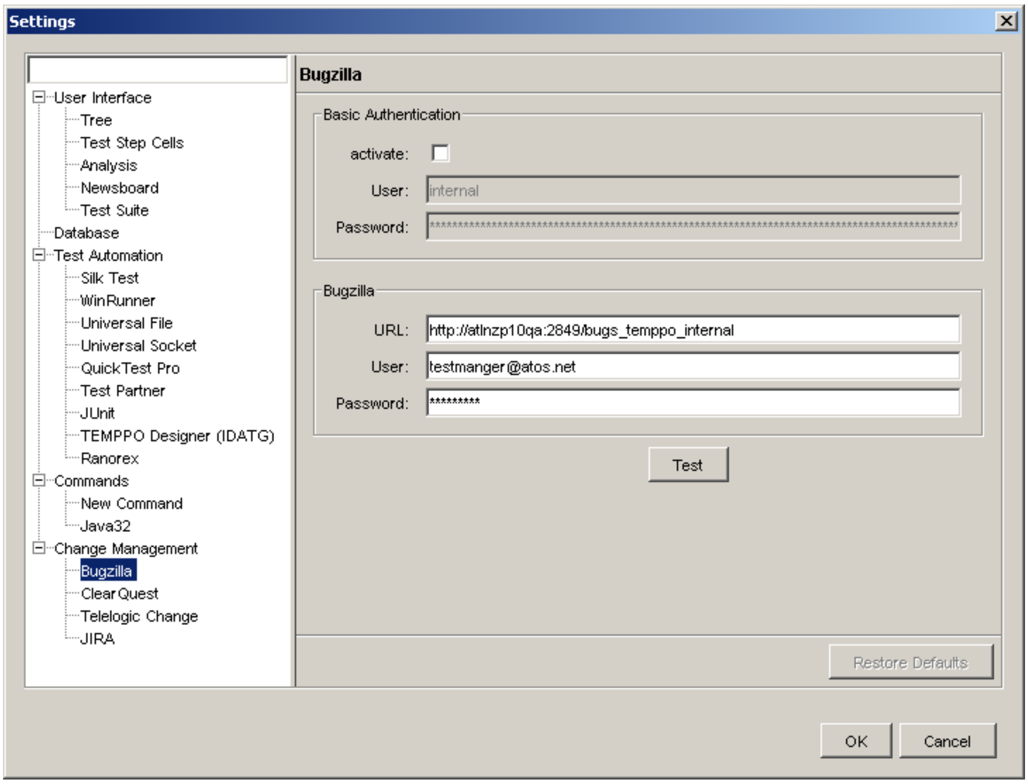
● Bugzilla
○ Telelogic Change
○ ClearQuest
○ JIRA

Figure 296 – Settings: Change Management

3.3.20.2.12.1 Bugzilla

The following parameters have to be specified if using Bugzilla:

- **Basic Authentication**: If the checkbox is activated, a proxy server authentication (user, password) can be used.
- **URL**: The http address of your Bugzilla server.
- **User**: User name who will post the bug
- **Password**: Password of user



Bugzilla

Basic Authentication

activate: □

User: internal

Password: **

Bugzilla

URL: http://atlnzp10qa:2849/bugs_temppo_internal

User: testmanger@atos.net

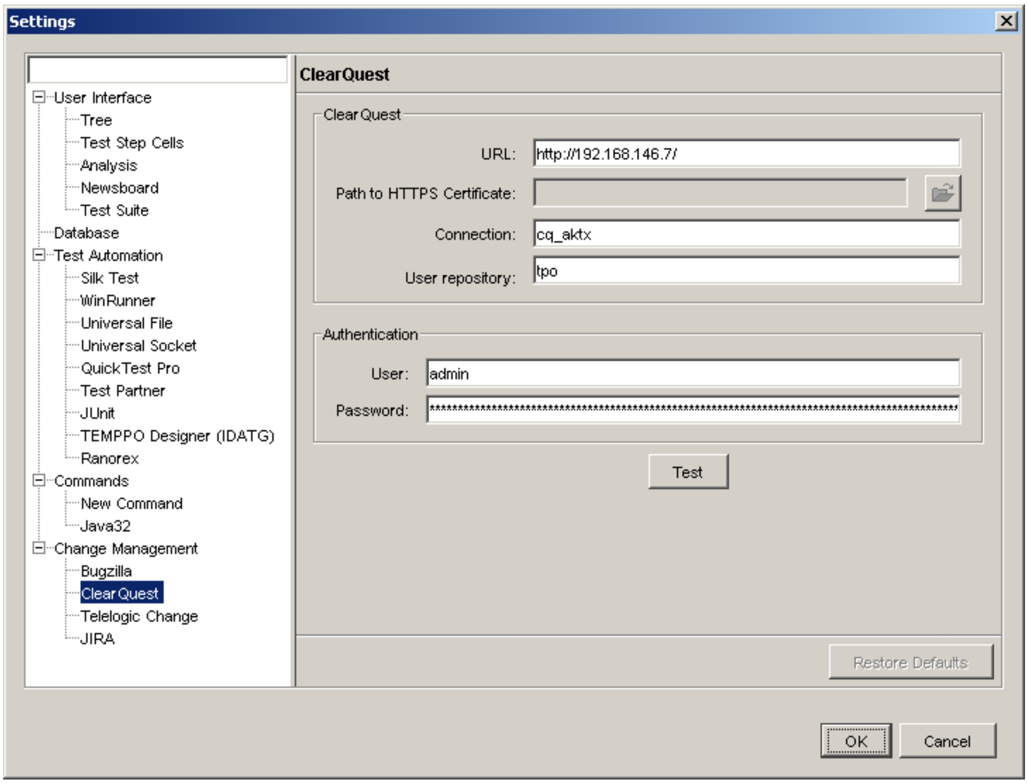
Password: *********

Figure 297 – Settings: Bugzilla

3.3.20.2.12.2 IBM Rational Clearquest

The following table shows the parameter matching of TEMPPO and Clearquest:

TEMPPO	Clearquest
URL	Web address of Cleaquest for login
Connection	Database
User Repository	Schema repository
User	User name
Password	Password



Settings

- User Interface
  - Tree
  - Test Step Cells
  - Analysis
  - Newsboard
  - Test Suite
- Database
- Test Automation
  - Silk Test
  - WinRunner
  - Universal File
  - Universal Socket
  - QuickTest Pro
  - Test Partner
  - JUnit
  - TEMPPO Designer (IDATG)
  - Ranorex
- Commands
  - New Command
  - Java32
- Change Management
  - Bugzilla
  - ClearQuest
  - Telelogic Change
  - JIRA

ClearQuest

ClearQuest

URL: http://192.168.146.7/

Path to HTTPS Certificate:

Connection: cq_aktx

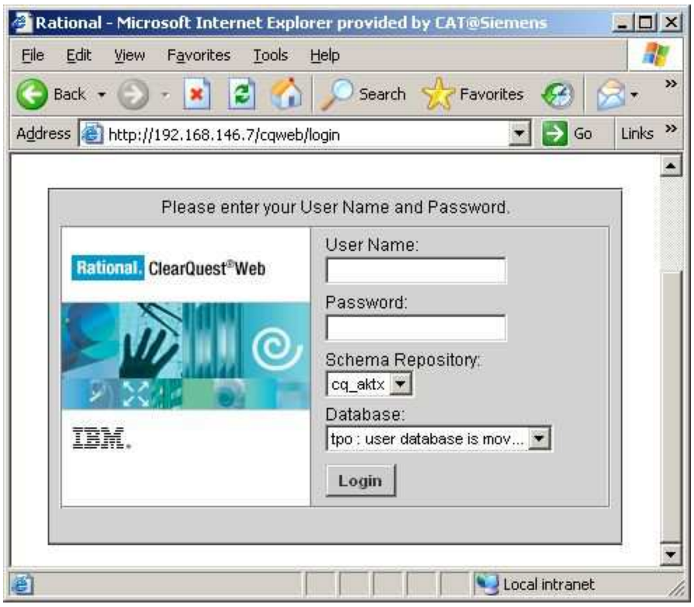
User repository: tpo

Authentication

User: admin

Password: **

Figure 298 – Settings: IBM Rational Clearquest



Rational - Microsoft Internet Explorer provided by CAT@Siemens
File Edit View Favorites Tools Help
Back
Search
Favorites
Go
Links
Rational. ClearQuest®Web
IBM.
Login
Local intranet

Address: http://192.168.146.7/cqweb/login

Please enter your User Name and Password.

User Name:

Password:

Schema Repository: cq_aktx

Database: tpo : user database is mov...

Figure 299 – Login to Clearquest

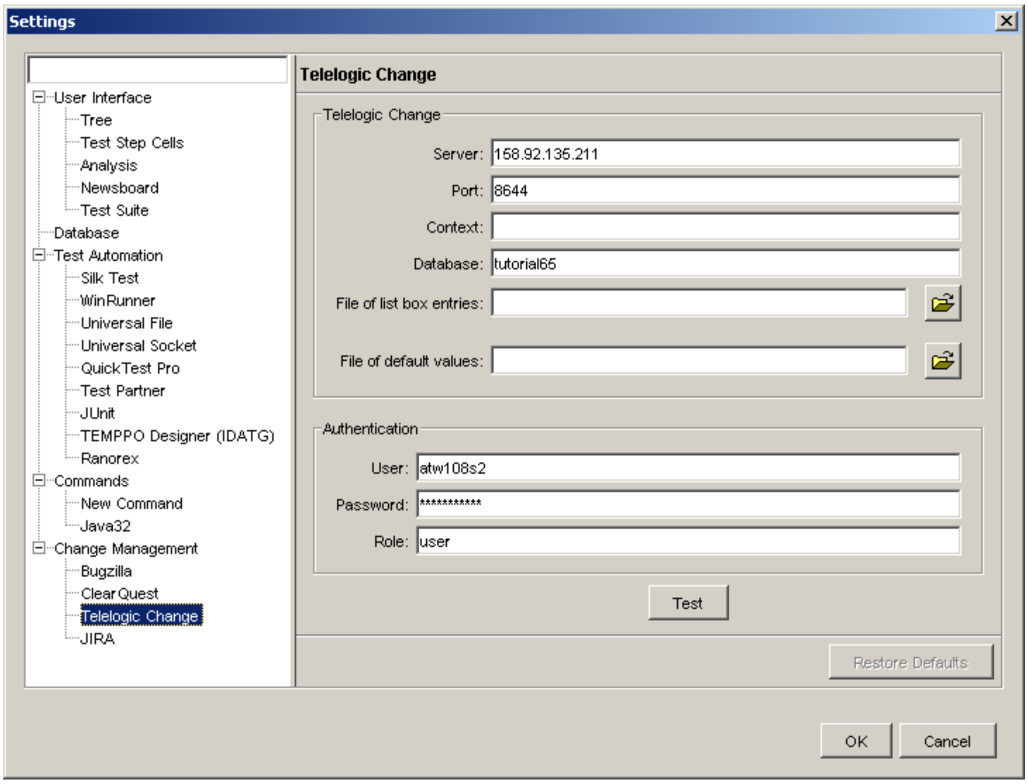
3.3.20.2.12.3 IBM Telelogic Change

The following parameters have to be specified if using Telelogic Change:

- **Server**, **Port**: IP address or server name, where Telelogic Change is running.
- **Context**: If your Telelogic Change is located at context, you have to add it here.
- **Database**: only the name of the database
- **File of list box entries**: path to the config-file of list boxes with its dependencies.

- **User**, **Password**: User name who will post a bug, and password of the user.
- **Role**: Role of the user in Telelogic Change

Example: If the web address of your Telelogic Change is located at http://158.92.135.211/change, it has to be separated like in the following screen shot:



Telelogic Change

Telelogic Change

Server: 158.92.135.211

Port: 8644

Context:

Database: tutorial65

File of list box entries:

File of default values:

Authentication

User: atw108s2

Password: ***********

Role: user

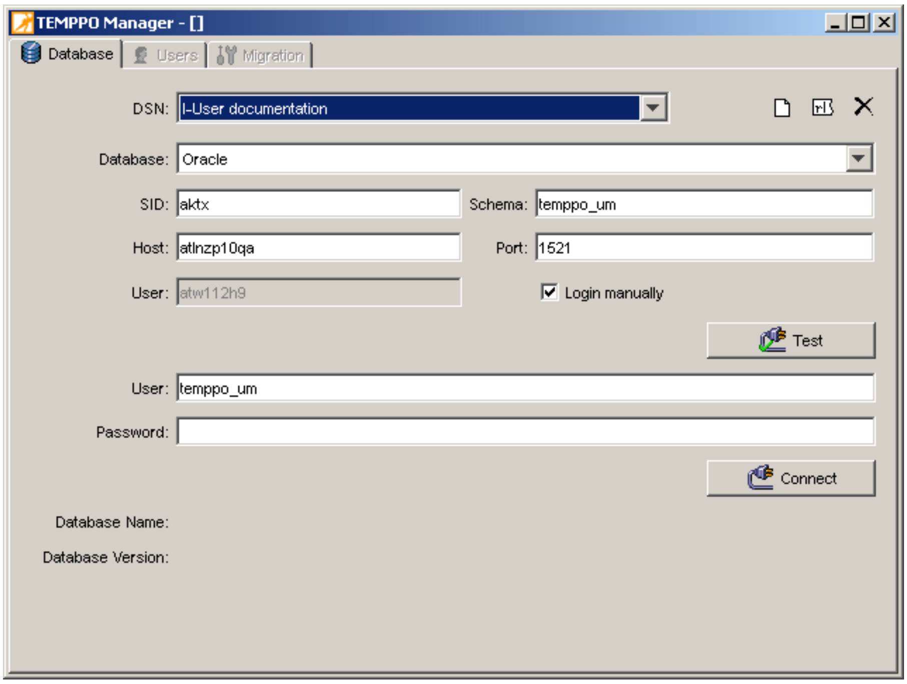
Figure 300 – Telelogic Change

4 TEMPPO Manager

The following TEMPPO configurations are done with the TEMPPO Manager tool:

- Database connection
- User management (create a TEMPPO super user
- Database migration

4.1 Database Connection



DSN: I-User documentation

Database: Oracle

SID: aktx

Schema: temppo_um

Host: atlnzp10qa

Port: 1521

User: atw112h9

☑ Login manually

User: temppo_um

Password:

Database Name:

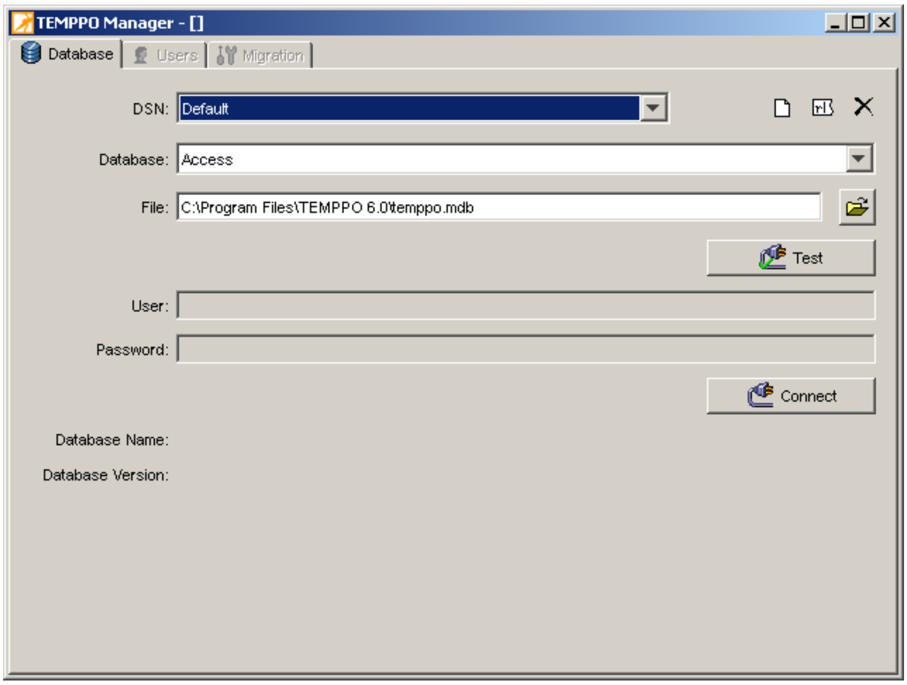
Database Version:

Figure 301: Manager – Database (Oracle)

In this tab you can define various database connection configurations, where each is identified by a **DSN** (Data Source Name). A new DSN can be created using the 🗋 button, which will prompt you to enter a unique DSN. A DSN can be deleted by clicking the ✕ button. A DSN can be renamed by clicking the ⊞ button. If you delete all DSNs, TEMPPO will not be able to connect to a database! For each DSN you have to specify several connection parameters, which are explained in the following.

TEMPPO supports two database management systems, selectable in the field database. Depending on the chosen one, you must specify several connection parameters:

- *Access:* For Access you just simply have to specify the path to Access mdb. TEMPPO automatically creates an ODBC entry.



DSN: Default

Database: Access

File: C:\Program Files\TEMPPO 6.0\temppo.mdb

User:

Password:

Database Name:

Database Version:

–

Figure 302 – Manager – Database (Access)

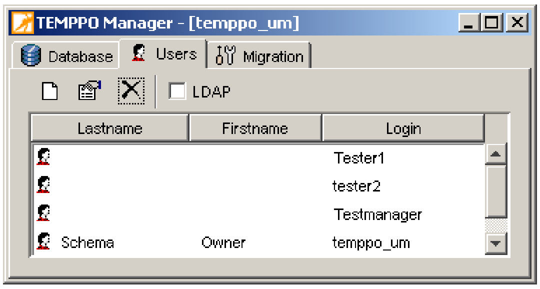
– *Oracle, MS SQL Server:* You must specify the database name (SID), the schema name (ask your database admin), the name or IP address of the database server (Host) and the port number (Port) at which the database server is listening for incoming connections. Contact your database administrator for this information.

The currently selected DSN is the active one, i.e. TEMPPO uses its connection parameters to connect to a database.

Using button **Test** you can verify your settings. A message will inform about the result of the verification.

By clicking **Connect** the TEMPPO Manager will establish a connection to the database (user and password are required for Oracle). This will cause the tabs Users and Migration getting enabled. Information about the database will be displayed on the bottom of the tab.

4.2 User management



□ LDAP

Lastname	Firstname	Login
		Tester1
		tester2
		Testmanager
Schema	Owner	temppo_um

Figure 303: Manager – User management

This tab is only active, if the test manager is connected to the database (see Database Connection). Users are created, changed and deleted here. The list shows all currently existing users.

User management in TEMPPO Manager is used to create the first super user, who has afterwards the rights to do all the administrative work in TEMPPO Administrator without any database password.

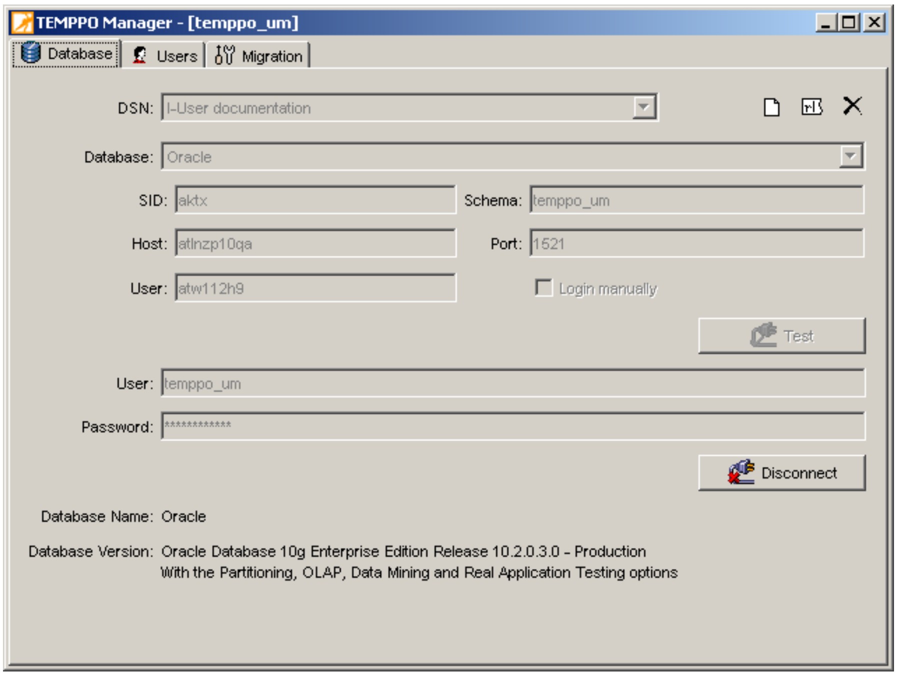
SQLSERVER: if a superuser is created, the following procedure has to be executed via a console or EnterpriseManager by an sysadmin: exec sp_addsrvrolemember N'$(sid)',N'sysadmin'

The first step is to create a super user, that can either be with MS Window login or a self created one with password.

It is not allowed to start TEMPPO twice with the same login. This could cause inconsistencies in the database

4.2.1 MS Window login

For creating a (super) user with MS Windows login, you have to connect to the database without activating the checkbox **Login manually**.



DSN: I-User documentation

Database: Oracle

SID: aktx
Schema: temppo_um

Host: atlnzp10qa
Port: 1521

User: atw112h9
□ Login manually

User: temppo_um

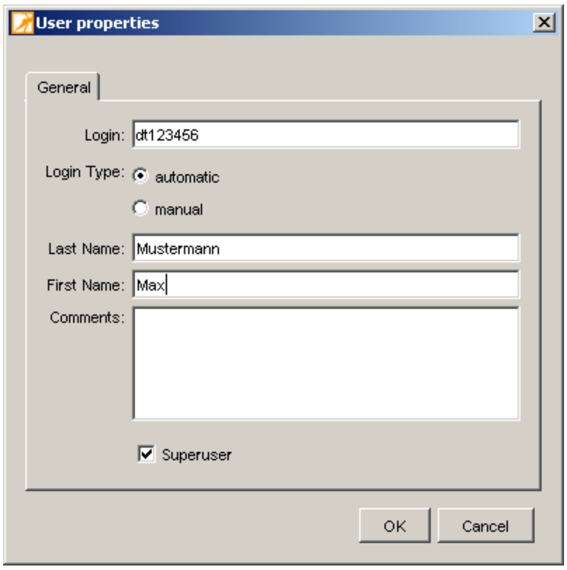
Password: ************

Database Name: Oracle

Database Version: Oracle Database 10g Enterprise Edition Release 10.2.0.3.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

Figure 304 – DB connect to create MS Window Login

When pressing the button **Connect**, Figure 303 is displayed. A new (super) user can be created by clicking the button **Create new user**. The input fields are explained in chapter 4.2.3.



User properties

General

Login: dt123456

Login Type: ● automatic ○ manual

Last Name: Mustermann

First Name: Max

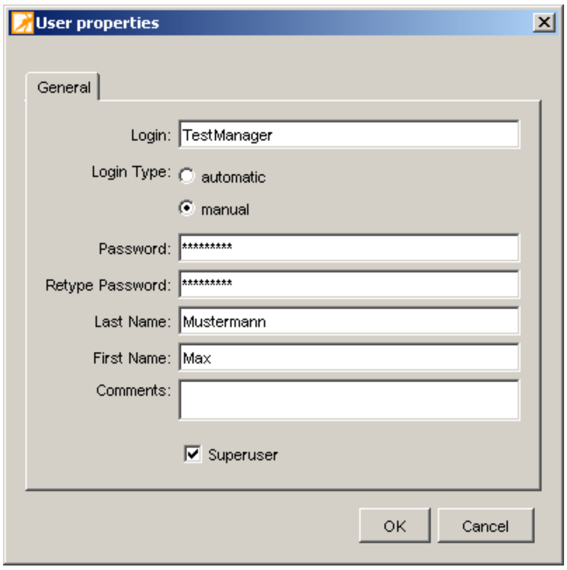
Comments:

☑ Superuser

Figure 305 – Create MS Window login

4.2.2 Login with password

For creating a (super) user with manual login, you have to choose the radio button **manual.** The input fields are explained in chapter 4.2.3. If you fill-in also **Last** and **First** name, they will be displayed in TEMPPO Test Manager instead of the **Login**.



User properties

General

Login: TestManager

Login Type: ○ automatic ● manual

Password: *********

Retype Password: *********

Last Name: Mustermann

First Name: Max

Comments:

☑ Superuser

Figure 306 – Create manual login

4.2.3 User Properties

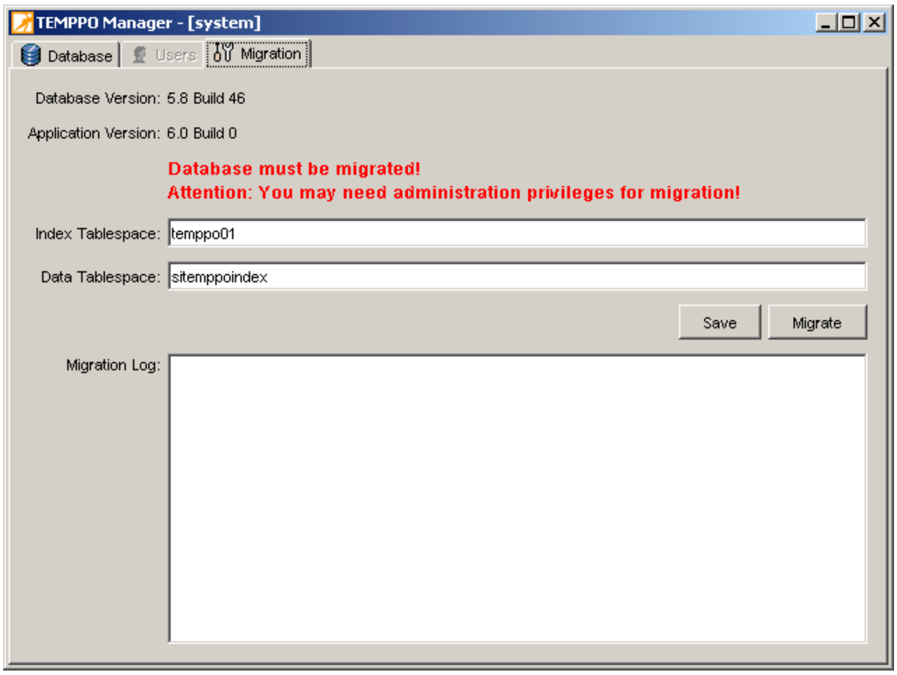
For specifying a user the following data has to be entered:

- **Login**: MS Windows(NT, XP, 2000, 7) login or manual one
- **Login Type**: automatic or manual
- **Password**: In case of manual login the password
- **Retype Password**: Retyping the password
- **Last name**: Will be shown in TEMPPO instead of login

- **First name**: Will be shown together with last name in TEMPPO
- **Comment**: Text field (optional)
- **Superuser**: Check Box for the role Superuser, can only be assigned in TEMPPO Manager

ⓘ **At least one TEMPPO Superuser has to be defined.**

4.3 Database migration



Database Version: 5.8 Build 46

Application Version: 6.0 Build 0

Database must be migrated!
Attention: You may need administration privileges for migration!

Index Tablespace: temppo01

Data Tablespace: sitemppoindex

Migration Log:

Figure 307: Manager – Database migration

This tab is only active, if the Administrator is connected to the database (see Database Connection). Database migration due to version changes of TEMPPO is done here. If migration is needed, the buttons are active. **Migrate** processes the built in migration script, with button **Save** the migration script can be saved to a file (SQL-script for Oracle, VB-Script for Access). The saved script must then be executed outside of TEMPPO.

5 TEMPPO Administrator

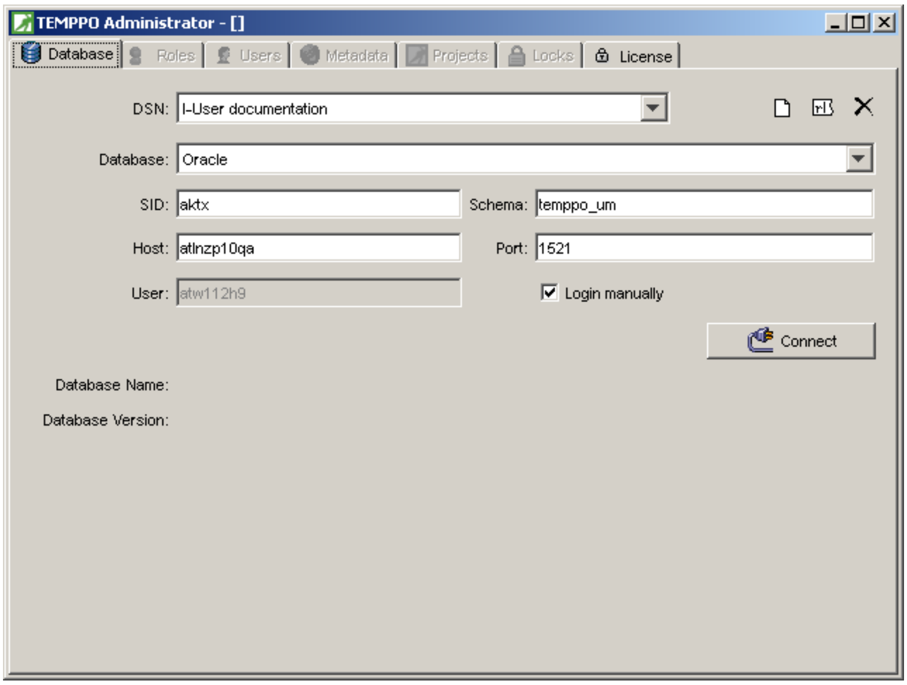
The following configuration settings are defined with the TEMPPO Administrator tool:

- Database connection
- Test automation settings
- Roles
- User management (except TEMPPO Superuser)
- Metadata
- Projects
- License server connection

ⓘ **Configuration settings are written into the admin.properties file, which is read by TEMPPO on application startup. Therefore TEMPPO must be restarted, before any configuration changes take effect.**

ⓘ **You may start more than one TEMPPO Administrator application. However: take into account that all Administrator applications write to the same admin.properties file.**

5.1 Database Connection



DSN: I-User documentation

Database: Oracle

SID: aktx

Schema: temppo_um

Host: atlnzp10qa

Port: 1521

User: atw112h9

☑ Login manually

Database Name:

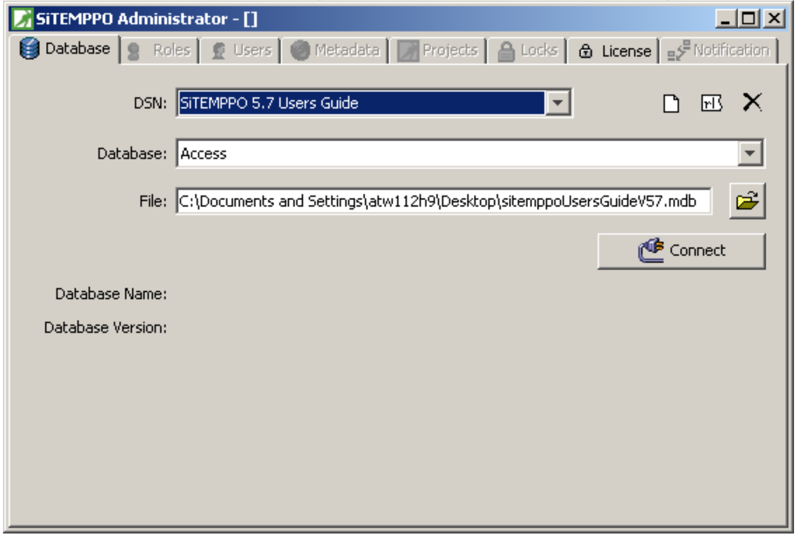
Database Version:

Figure 308: Administrator – Database (Oracle)

In this tab you can define various database connection configurations, where each is identified by a **DSN** (Data Source Name). A new DSN can be created using the 🗋 button, which will prompt you to enter a unique DSN. A DSN can be deleted by clicking the ✕ button. A DSN can be renamed by clicking the ⎘ button. If you delete all DSNs, TEMPPO will not be able to connect to a database! For each DSN you have to specify several connection parameters, which are explained in the following.

TEMPPO supports two database management systems, selectable in the field database. Depending on the chosen one, you must specify several connection parameters:

- *Access:* For Access you just simply have to specify the path to Access mdb. TEMPPO automatically creates an ODBC entry.



DSN: SiTEMPPO 5.7 Users Guide

Database: Access

File: C:\Documents and Settings\atw112h9\Desktop\sitemppoUsersGuideV57.mdb

Database Name:

Database Version:

–

Figure 309 – Administrator – Database (Access)

- *Oracle, MS SQL Server*: You must specify the database name (SID), the schema name (ask your database admin), the name or IP address of the database server (Host) and the port number (Port) at which the database server is listening for incoming connections. Contact your database administrator for this information.

  Furthermore you can activate the checkbox **Login manually**, that activates the manual login window before displaying the TEMPPO main window.

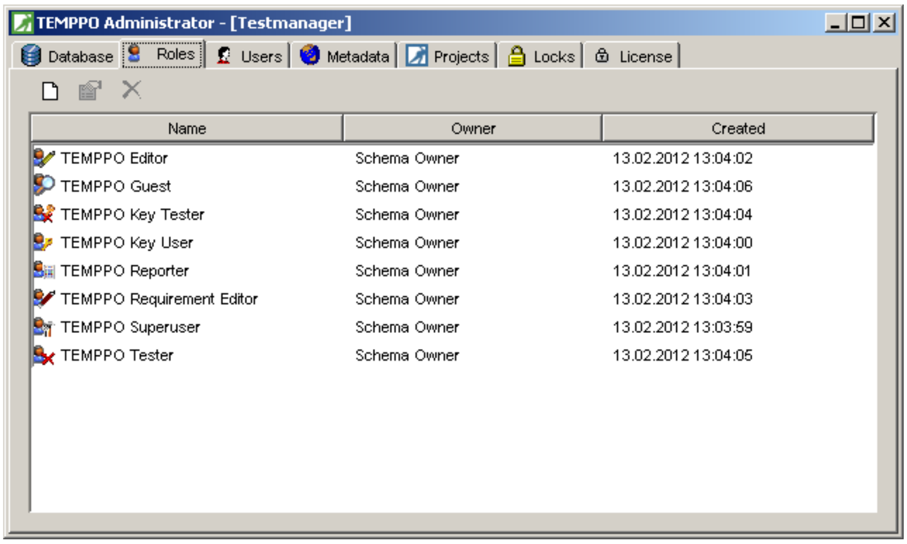
The currently selected DSN is the active one, i.e. TEMPPO uses its connection parameters to connect to a database.

Using button **Test** you can verify your settings. A message will inform about the result of the verification.

By clicking **Connect** the TEMPPO Administrator will establish a connection to the database (user and password are required for Oracle). This will cause the tabs Users, Projects, Migration and Roles getting enabled (if the user logged in is allowed to do so by his assigned roles. Information about the database will be displayed in the bottom of the tab.

5.2 Roles

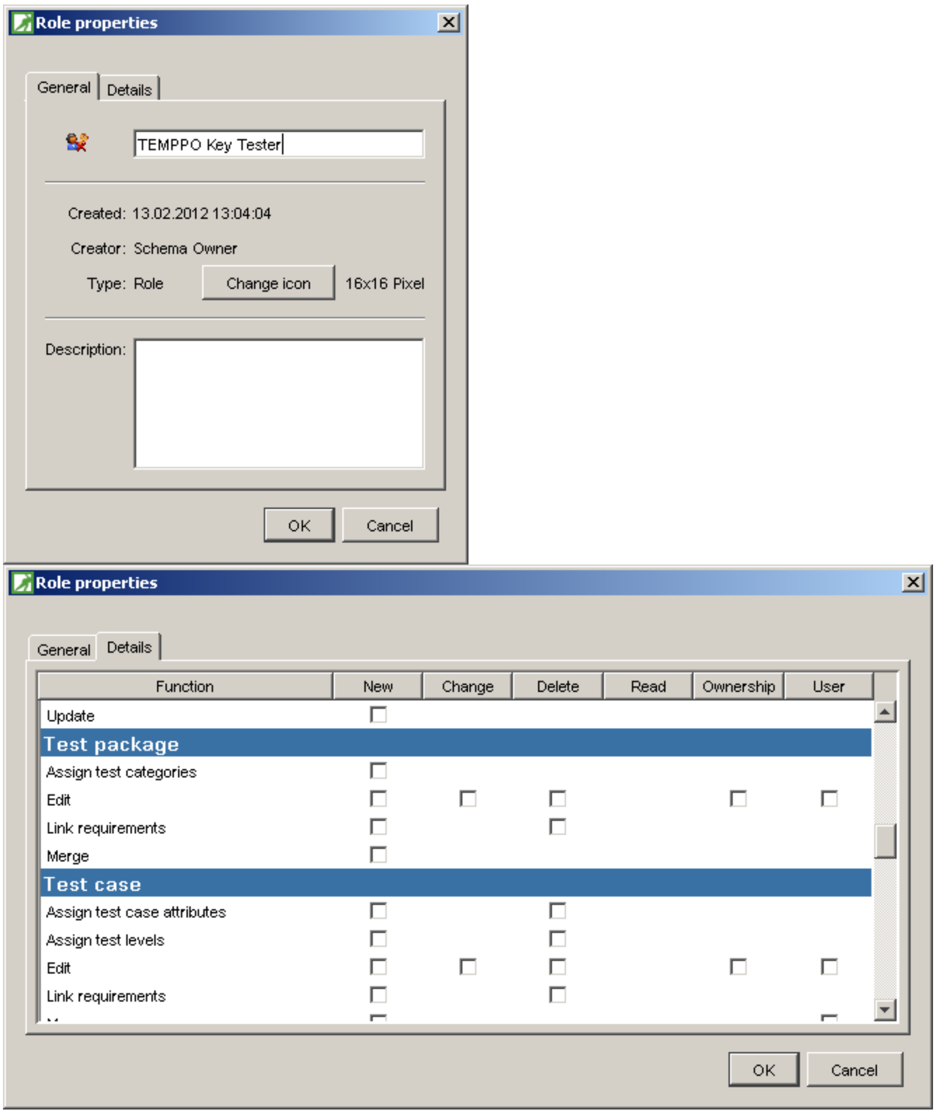
The access to TEMPPO functionality is managed by roles. In the following tab **Roles** the predefined TEMPPO roles are listed. On the one hand roles can be edited or adapted and on the other side new roles can be created.



Name	Owner	Created
TEMPPO Editor	Schema Owner	13.02.2012 13:04:02
TEMPPO Guest	Schema Owner	13.02.2012 13:04:06
TEMPPO Key Tester	Schema Owner	13.02.2012 13:04:04
TEMPPO Key User	Schema Owner	13.02.2012 13:04:00
TEMPPO Reporter	Schema Owner	13.02.2012 13:04:01
TEMPPO Requirement Editor	Schema Owner	13.02.2012 13:04:03
TEMPPO Superuser	Schema Owner	13.02.2012 13:03:59
TEMPPO Tester	Schema Owner	13.02.2012 13:04:05

Figure 310 – Roles

Adding, editing and deleting are allowed. In the tab **General** you can fill in the name and description or change the icon. In the tab **Details** the rights are listed.



TEMPPO Key Tester

Created: 13.02.2012 13:04:04

Creator: Schema Owner

Type: Role

16x16 Pixel

Description:

Function	New	Change	Delete	Read	Ownership	User
Update	□					
Test package						
Assign test categories	□					
Edit	□	□	□		□	□
Link requirements	□		□			
Merge	□					
Test case						
Assign test case attributes	□		□			
Assign test levels	□		□			
Edit	□	□	□		□	□
Link requirements	□		□			

Figure 311 – Role properties: General/ Details

The functions are predefined and cover the whole functionality of TEMPPO Administrator and TEMPPO.

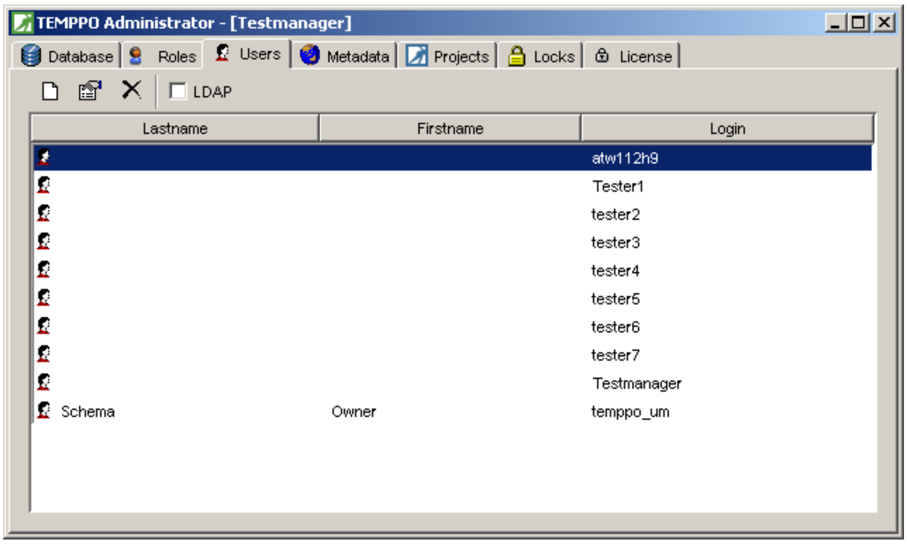
Level of access of the function:

- New (or execute)
- Edit
- Delete
- Read
- Move Ownership
- Consider User (objects can only be deleted if the user is the owner of the object)

In the dialog only the possible checkboxes are shown. E.g. report: It only makes sense and is only possible to execute a report. Therefore only the checkbox **New** (or **execute**) can be selected.

By this way roles can be created and adapted due to user demands.

5.3 User management



□ LDAP

Lastname	Firstname	Login
		atw112h9
		Tester1
		tester2
		tester3
		tester4
		tester5
		tester6
		tester7
		Testmanager
Schema	Owner	temppo_um

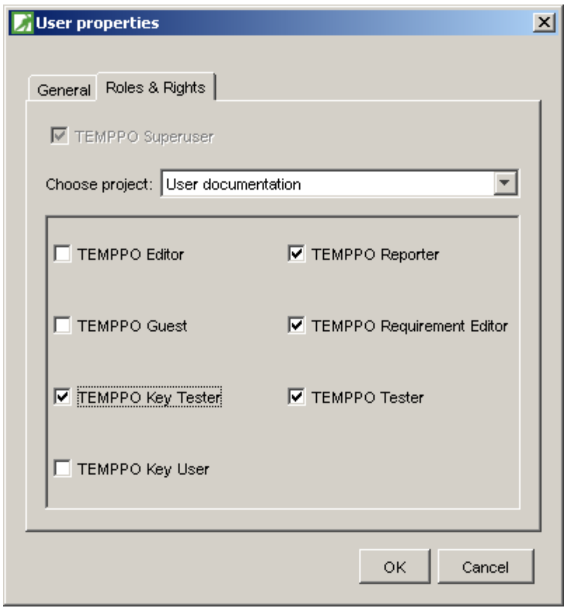
Figure 312: Administration – User management

This tab is only active, if the connected user is allowed to by his assigned roles. Users are created, changed and deleted here. The list shows all currently existing users. By clicking the **New** or **Properties** button, the user properties dialog pops up, where you can change the user name. Clicking the **Delete** button will delete the selected user after you confirmed the verification message.

By selecting the checkbox **LDAP**, you can connect to a **LDAP server** and import or synchronize users.

In principal, it is possible to use 2 different logins, on the one hand user can login to TEMPPO automatically with their MS Windows account and on the other side you can create user logins named as you like. Such "manual" logins need a password.

The same functions with the exception of role assignment are offered in TEMPPO Manager, for details see chapter 4.2,



☑ TEMPPO Superuser

Choose project: User documentation

□ TEMPPO Editor ☑ TEMPPO Reporter

□ TEMPPO Guest ☑ TEMPPO Requirement Editor

☑ TEMPPO Key Tester ☑ TEMPPO Tester

□ TEMPPO Key User

Figure 313: User properties- Roles & Rights

To assign a defined role to a user, select tab Roles & Right and select a project. Then check the roles the user should get for the chosen project and press the button “apply”. Choose another project and assign roles to the user by clicking the required checkboxes and pressing “apply”. A user can have different roles in different projects.

ⓘ **The TEMPPO Superuser role can only be assigned in the TEMPPO Manager – Users tab.**

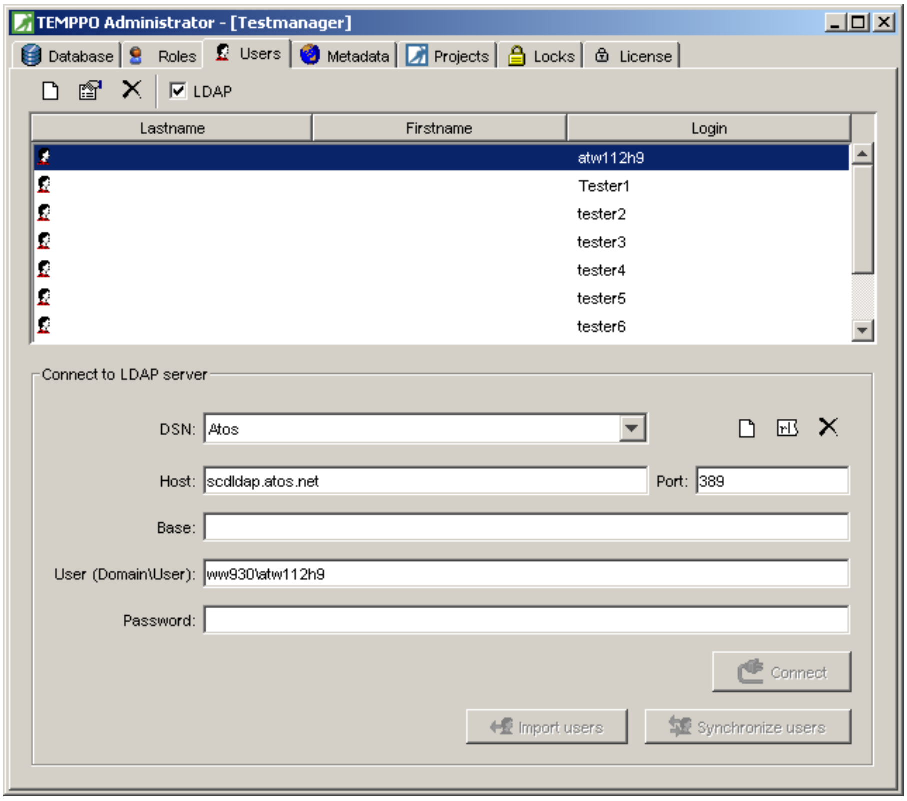
If a new role is assigned to a user, the user has to restart TEMPPO to have access to the functionality provided by the newly assigned role.

5.3.1 LDAP

By selecting the checkbox LDAP a field for configuring LDAP connections is shown (see Figure 244). After a successful connect you can import users from a LDAP server or you can synchronize the existing TEMPPO users with the LDAP server.

For imported users you can see the LDAP server from which the user was imported. If a user was created manually, the value in the column LDAP server is not set.

5.3.1.1 LDAP Connection



☑ LDAP

Lastname	Firstname	Login
		atw112h9
		Tester1
		tester2
		tester3
		tester4
		tester5
		tester6

Connect to LDAP server

DSN: Atos

Host: scdldap.atos.net Port: 389

Base:

User (Domain\User): ww930\atw112h9

Password:

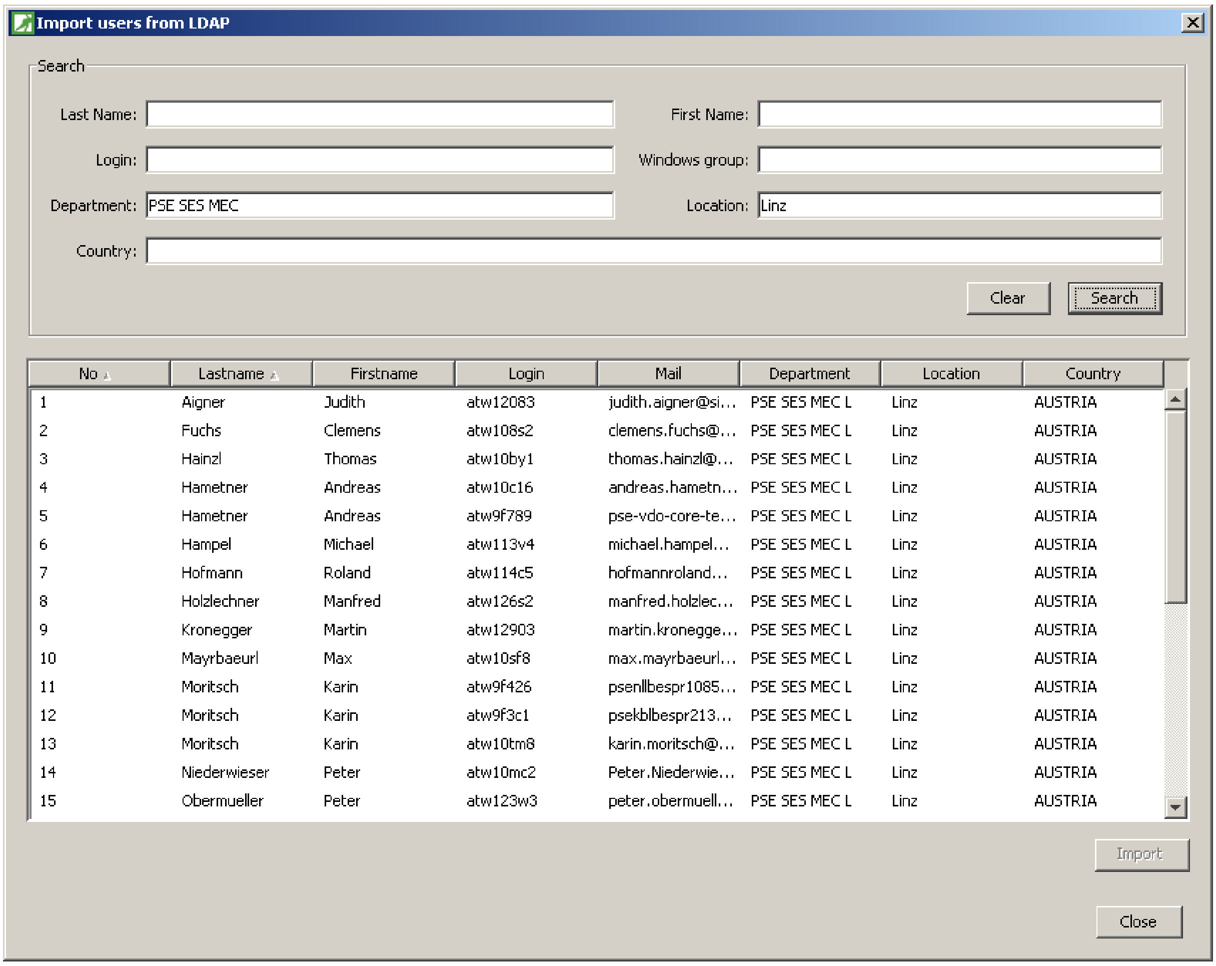
Figure 314: Connection to the LDAP server

You can define various LDAP connection configurations, where each is identified by a DSN (Data Source Name). A new DSN can be created using the 🗋 button, which will prompt you to enter a unique DSN. A DSN can be deleted by clicking the ✕ button. A DSN can be renamed by clicking the ▭ button. For each DSN you have to specify several connection parameters, which are explained in the following.

You must specify the name of the server (**Host**), the port number (**Port**), the domain name (**Base**) and the user data (**User** and **Password**) to connect to the LDAP server. The login name must be specified in the form of Domain\User.

By clicking **Connect** the TEMPPO Administrator will establish a connection to the LDAP server. If the connect to the LDAP server was successful the buttons **Import users** and **Synchronize users** are enabled. You can import users by clicking the button **Import users**. By clicking the button **Synchronize users** you can synchronize the existing TEMPPO users with the LDAP server.

5.3.1.2 Importing users from LDAP



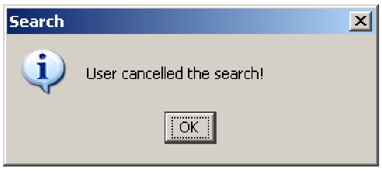
Last Name:
First Name:
Login:
Windows group:
Department: PSE SES MEC
Location: Linz
Country:

No	Lastname	Firstname	Login	Mail	Department	Location	Country
1	Aigner	Judith	atw12083	judith.aigner@si...	PSE SES MEC L	Linz	AUSTRIA
2	Fuchs	Clemens	atw108s2	clemens.fuchs@...	PSE SES MEC L	Linz	AUSTRIA
3	Hainzl	Thomas	atw10by1	thomas.hainzl@...	PSE SES MEC L	Linz	AUSTRIA
4	Hametner	Andreas	atw10c16	andreas.hametn...	PSE SES MEC L	Linz	AUSTRIA
5	Hametner	Andreas	atw9f789	pse-vdo-core-te...	PSE SES MEC L	Linz	AUSTRIA
6	Hampel	Michael	atw113v4	michael.hampel...	PSE SES MEC L	Linz	AUSTRIA
7	Hofmann	Roland	atw114c5	hofmannroland...	PSE SES MEC L	Linz	AUSTRIA
8	Holzlechner	Manfred	atw126s2	manfred.holzlec...	PSE SES MEC L	Linz	AUSTRIA
9	Kronegger	Martin	atw12903	martin.kronegge...	PSE SES MEC L	Linz	AUSTRIA
10	Mayrbaeurl	Max	atw10sf8	max.mayrbaeurl...	PSE SES MEC L	Linz	AUSTRIA
11	Moritsch	Karin	atw9f426	psenllbespr1085...	PSE SES MEC L	Linz	AUSTRIA
12	Moritsch	Karin	atw9f3c1	psekblbespr213...	PSE SES MEC L	Linz	AUSTRIA
13	Moritsch	Karin	atw10tm8	karin.moritsch@...	PSE SES MEC L	Linz	AUSTRIA
14	Niederwieser	Peter	atw10mc2	Peter.Niederwie...	PSE SES MEC L	Linz	AUSTRIA
15	Obermueller	Peter	atw123w3	peter.obermuell...	PSE SES MEC L	Linz	AUSTRIA

Figure 315: Importing users from LDAP

You can search for a last name, first name, login, windows group, department, location and a country. By clicking the button **Clear** you can delete the search preferences. Clicking the **Search** button will perform the search and the result is shown in the table below.

The search can be cancelled at any time by clicking the button **Cancel**.



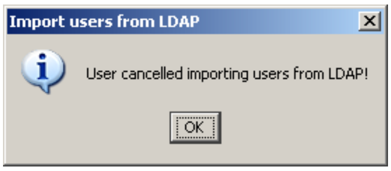
User cancelled the search!

Figure 316: Cancel a LDAP search

ⓘ **If you are searching for a Windows group or if the search output is very large, the result of the search might not be complete, because the performance of the LDAP server varies. So if you are searching for a specific user, try to find him by searching for the department or other criterias.**

After selecting one or several users in the table the button **Import** is enabled. By clicking this button you can import the selected users into TEMPPO. If you select a user that is already a TEMPPO user, the user is updated.

You can cancel the import at any time by clicking the button **Cancel**.


Import users from LDAP

User cancelled importing users from LDAP!

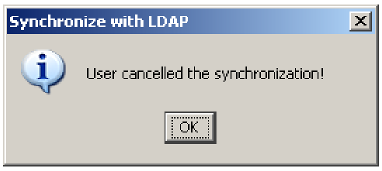
Figure 317: Cancel the import from LDAP

5.3.1.3 Synchronizing users with LDAP

A LDAP search for all TEMPPO users is performed and if they were found the first name and the last name are updated.

ⓘ **The synchronization will be performed for each user from the current LDAP server or for manually created users. If a manually created user was found, the column LDAP server is set.**

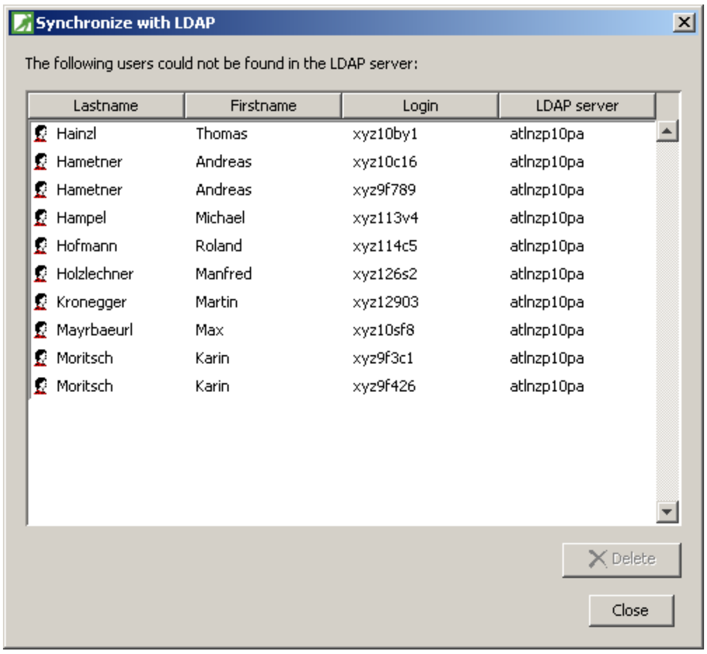
The synchronization can be cancelled at any time by clicking the button **Cancel**.


Synchronize with LDAP

User cancelled the synchronization!

Figure 318: Cancel the synchronization with LDAP

If there are TEMPPO users with the current LDAP server as origin that could not be found in the LDAP server, a dialog shows these users. You can delete users by clicking the button **Delete**.



The following users could not be found in the LDAP server:

Lastname	Firstname	Login	LDAP server
Hainzl	Thomas	xyz10by1	atlnzp10pa
Hametner	Andreas	xyz10c16	atlnzp10pa
Hametner	Andreas	xyz9f789	atlnzp10pa
Hampel	Michael	xyz113v4	atlnzp10pa
Hofmann	Roland	xyz114c5	atlnzp10pa
Holzlechner	Manfred	xyz126s2	atlnzp10pa
Kronegger	Martin	xyz12903	atlnzp10pa
Mayrbaeurl	Max	xyz10sf8	atlnzp10pa
Moritsch	Karin	xyz9f3c1	atlnzp10pa
Moritsch	Karin	xyz9f426	atlnzp10pa

Figure 319: Result of the synchronization with LDAP

5.4 Metadata

You can customize Metadata for the whole database, which provides a pool of data that can be used in various projects. If you want to use special metadata in a project, you have to assign them (see chapter 5.5.5).

The tab **Metadata** is divided into 6 tabs (if the tabs are not enabled, you do not have the right to see it). The following values are predefined:

– Test Levels: module test, integration test, system test, acceptance test, regression test

– Test Categories: state-based, GUI-layout, task-based, performance, resource, security, safety, portability, reliability, maintenance, reuse, usability.

– Attributes: none

– Uploads: none

– Requirement Structure: none

– User Fields: none

These tabs are described in the following chapters.

5.4.1 General Behavior

- Metadata are database wide.
- Metadata in a project can only be used, if they are assigned to the project.
- Metadata names have to be unique within all projects.

5.4.1.1 New Metadata Entries

How to create a new metadata in TEMPPO?

First select the corresponding tab (test level, test category, attributes, uploads, requirements user fields) and click 🗋.

There are two possibilities now:

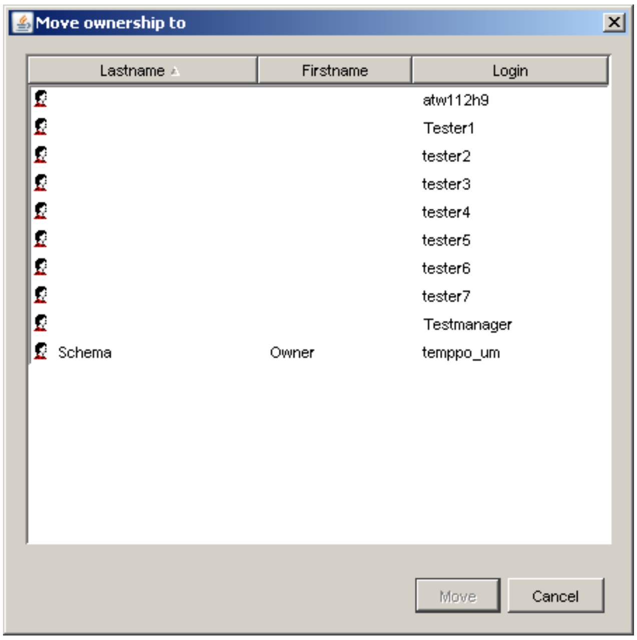
1) Type the name of the Metadata, the description and press the button OK. TEMPPO verifies now, if the Metadata already exists. If the metadata is existing, you are asked, if you want to assign this metadata to your project. If the metadata does not exist, it is saved to the metadata pool and the assignment to the project is made.
2) Type in a match criterion of the metadata and click on the arrow of the combo box. All metadata entries that match the criterion are listed in the combo box. Choose one, press the button ok and confirm the assignment question.

5.4.1.2 Delete Metadata Entries

Choose the right metadata tab and click the button Delete. If the metadata is not referenced in any project, the metadata is deleted.

5.4.1.3 Move ownership

In the TEMPPO Administrator there is the possibility to move the ownership of a metadata. After pressing the button "Move Ownership" the following dialog comes up.



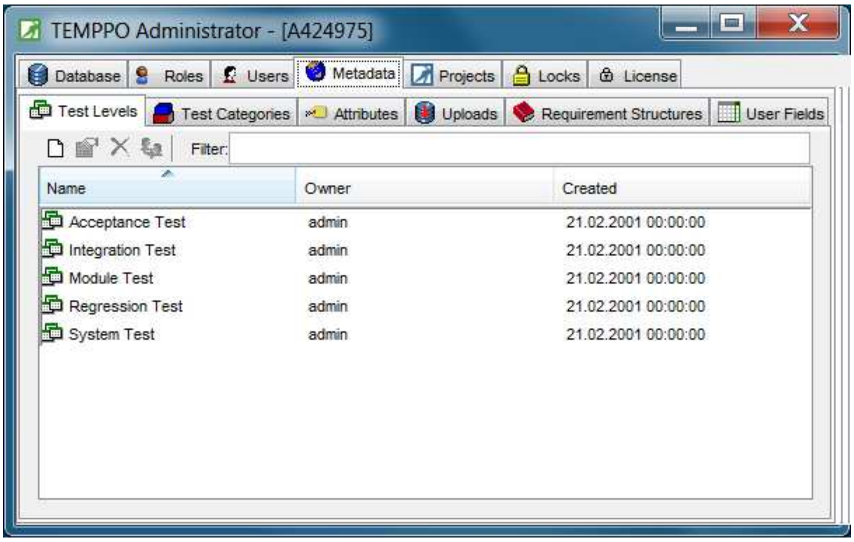
Lastname	Firstname	Login
		atw112h9
		Tester1
		tester2
		tester3
		tester4
		tester5
		tester6
		tester7
		Testmanager
Schema	Owner	temppo_um

Figure 320: Move Ownership

The new owner of the metadata can be chosen.

5.4.2 Test Levels

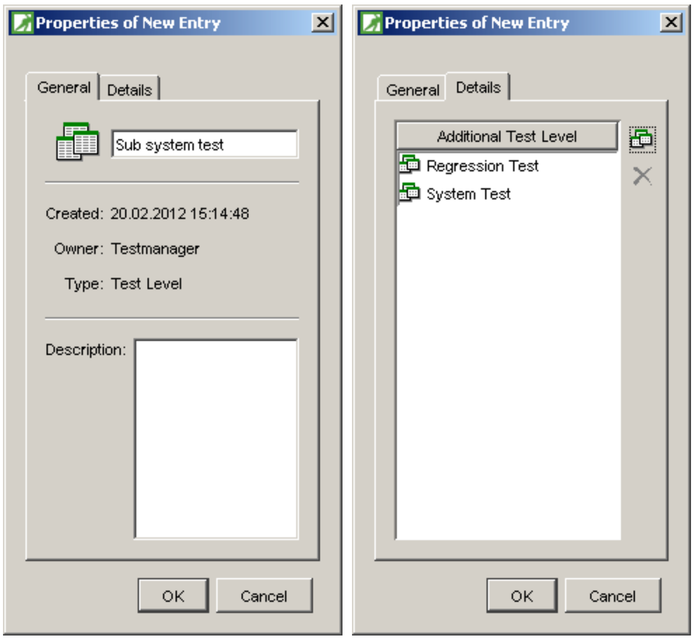
Here you can add, delete, view and filter test levels. In Figure 321 all test levels are displayed with name, owner and creation date.



Name	Owner	Created
Acceptance Test	admin	21.02.2001 00:00:00
Integration Test	admin	21.02.2001 00:00:00
Module Test	admin	21.02.2001 00:00:00
Regression Test	admin	21.02.2001 00:00:00
System Test	admin	21.02.2001 00:00:00

Figure 321 – Metadata: Test Levels

A new test level can be added by pressing the button **New** in the tab **Test levels** and Figure 322 is displayed.



Sub system test

Created: 20.02.2012 15:14:48

Owner: Testmanager

Type: Test Level

Regression Test

System Test

Figure 322 – New Test Level

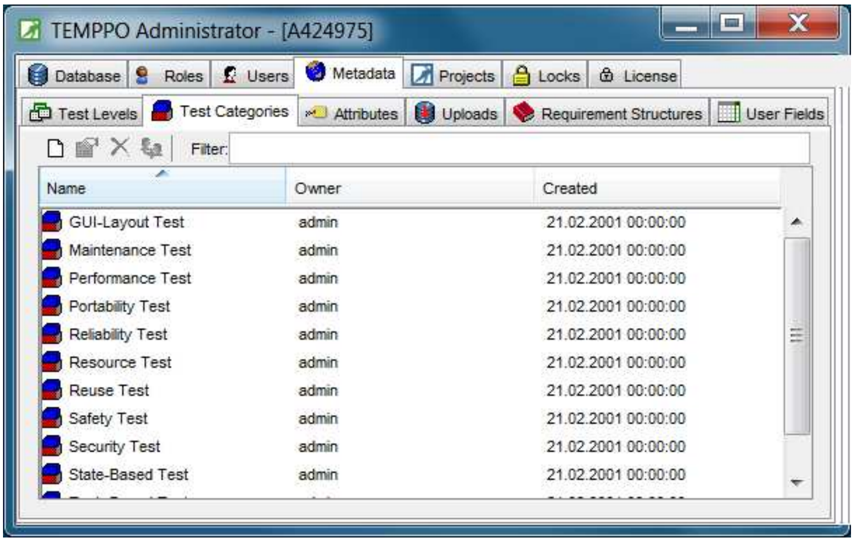
In this window general information like name, creation date and owner and a **Description** of the test level can be entered. In the second tab (**Details**) additional test levels can be added. **(Additional) test levels** are related to a test level.

Furthermore, it is possible to add supplementary test levels for the purpose of using test cases in different test levels. For example, there is a test structure related with test level System Test, which contains the additional test levels Acceptance and Regression Test. Each test case in this test structure can now be assigned to one or both of the additional test levels (see 3.1.2.4).

ⓘ **It's not possible to delete test levels, which are already used in projects (test structures or test cases).**

5.4.3 Test Categories

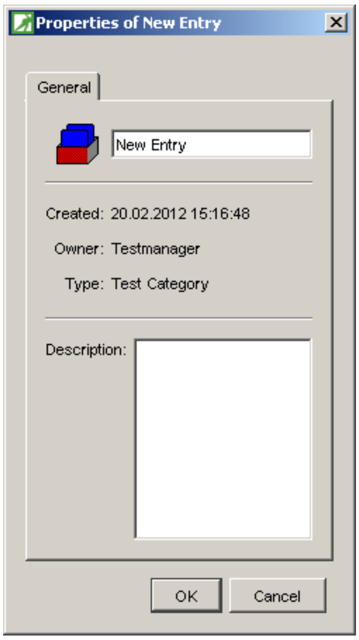
Test categories are used to denote test packages that contain only test cases of a special type.



Name	Owner	Created
GUI-Layout Test	admin	21.02.2001 00:00:00
Maintenance Test	admin	21.02.2001 00:00:00
Performance Test	admin	21.02.2001 00:00:00
Portability Test	admin	21.02.2001 00:00:00
Reliability Test	admin	21.02.2001 00:00:00
Resource Test	admin	21.02.2001 00:00:00
Reuse Test	admin	21.02.2001 00:00:00
Safety Test	admin	21.02.2001 00:00:00
Security Test	admin	21.02.2001 00:00:00
State-Based Test	admin	21.02.2001 00:00:00

Figure 323 – Metadata: Test Categories

Here you can add, delete, view and filter test categories. When pressing 🗋, the window **Test Category Properties** is opened. A name must be entered, and user name and creation date are suggested automatically. The same window is shown when applying 📝 for editing the test category properties.



New Entry

Created: 20.02.2012 15:16:48

Owner: Testmanager

Type: Test Category

Description:

Figure 324 – Test Category Properties

5.4.4 Attributes

Attributes are applied to test cases, test results and test suites.

They can be applied in addition to the standard attributes (owner, creation date, priority, and working state). Such attributes like working state, tester, sample phase etc. help the test manager to realize his objectives of the test plan.

Therefore it should be planned before creating a test structure or a test execution, which attributes are used.

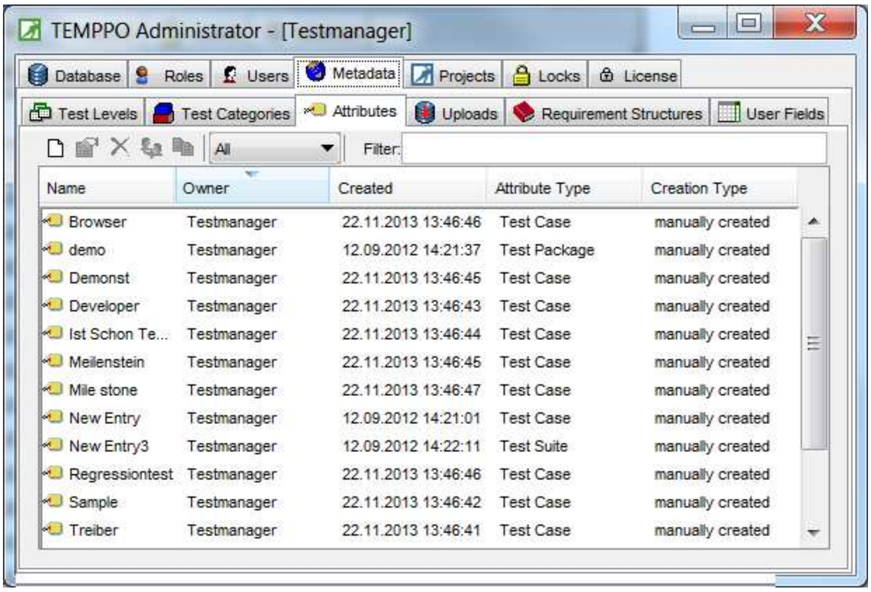
Especially for pursuing the test objectives it is important to define these attributes in TEMPPO. During the test process you may be asked always the same questions like:

– How is the progress of test case development in comparison to the last day/week/month?
– How many test cases are ready for test execution?
– How many test cases are ready for regression testing?
– How many test cases of the current test suite are passed/failed?

Besides test case attributes, you can assign attributes to test suites and test case results, too. These attributes can be used for different issues. E.g., during test case execution the version of the SUT (= software under test) sometimes changes. But the test manager wants to know which test cases were executed with which SUT version or which SUT versions are used in the test suite.

Moreover, predefined attributes provides the users from making mistakes when writing the necessary information.

Attributes are also used for filtering, analyzing and reporting the test structure or the test suite.



Name	Owner	Created	Attribute Type	Creation Type
Browser	Testmanager	22.11.2013 13:46:46	Test Case	manually created
demo	Testmanager	12.09.2012 14:21:37	Test Package	manually created
Demonst	Testmanager	22.11.2013 13:46:45	Test Case	manually created
Developer	Testmanager	22.11.2013 13:46:43	Test Case	manually created
Ist Schon Te...	Testmanager	22.11.2013 13:46:44	Test Case	manually created
Meilenstein	Testmanager	22.11.2013 13:46:45	Test Case	manually created
Mile stone	Testmanager	22.11.2013 13:46:47	Test Case	manually created
New Entry	Testmanager	12.09.2012 14:21:01	Test Case	manually created
New Entry3	Testmanager	12.09.2012 14:22:11	Test Suite	manually created
Regressiontest	Testmanager	22.11.2013 13:46:46	Test Case	manually created
Sample	Testmanager	22.11.2013 13:46:42	Test Case	manually created
Treiber	Testmanager	22.11.2013 13:46:41	Test Case	manually created

Figure 325 – Metadata: Attributes

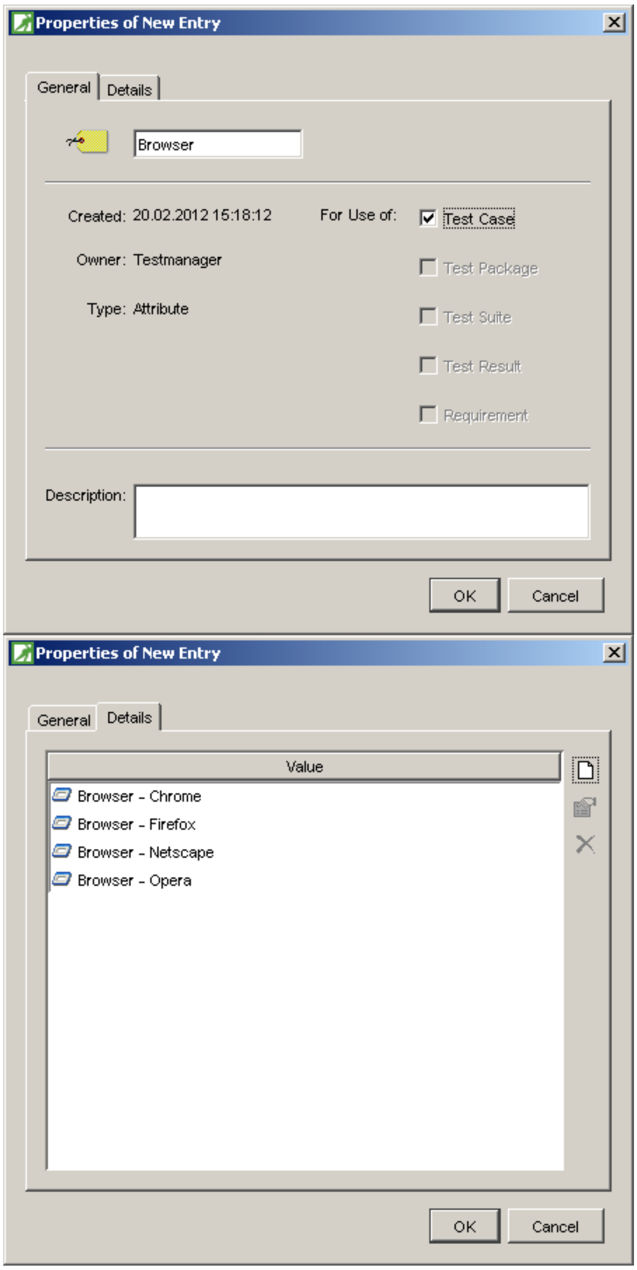
Here, you can add, delete, view and fiter attributes. In the master screen, the selection grouped by type can be changed by the box above.

When pressing🗋, the window **Attribute Properties** is opened. A name can be entered, and user name and creation date are suggested automatically. In addition, the attribute has to be categorized for a test case, test suite or/and test result. A test suite attribute may also be used for test execution attributes, but not for a test case's one.

This implies that the usage of an attribute can be changed after creation.

The boxes for categorizing are disabled, if the attribute is already used.

In tab **Details** the possible attribute values must be defined.



Properties of New Entry

Browser

Created: 20.02.2012 15:18:12

Owner: Testmanager

Type: Attribute

For Use of:

☑ Test Case

□ Test Package

□ Test Suite

□ Test Result

□ Requirement

Description:

Properties of New Entry

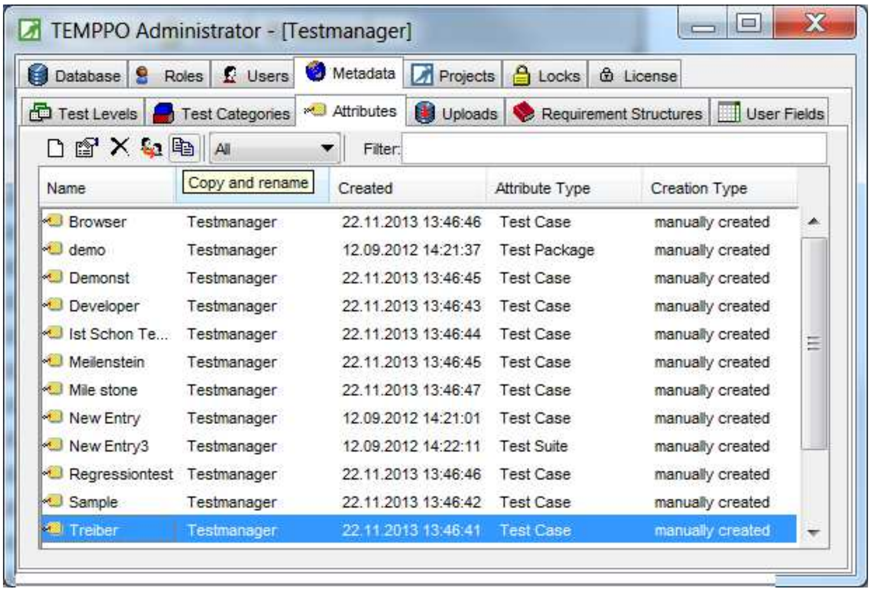
Value
Browser - Chrome
Browser - Firefox
Browser - Netscape
Browser - Opera

Figure 326 – Attribute Properties

ⓘ **An attribute cannot be deleted, if it's already used in a test case.**

5.4.4.1 Copy and rename of UDA

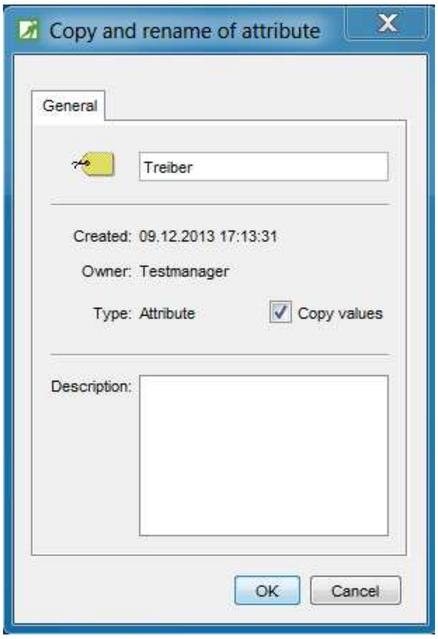
For user defined attributes the possibility of copy and rename (copy an existing UDA with or without values and rename it) is offered.



Name		Created	Attribute Type	Creation Type
Browser	Testmanager	22.11.2013 13:46:46	Test Case	manually created
demo	Testmanager	12.09.2012 14:21:37	Test Package	manually created
Demonst	Testmanager	22.11.2013 13:46:45	Test Case	manually created
Developer	Testmanager	22.11.2013 13:46:43	Test Case	manually created
Ist Schon Te...	Testmanager	22.11.2013 13:46:44	Test Case	manually created
Meilenstein	Testmanager	22.11.2013 13:46:45	Test Case	manually created
Mile stone	Testmanager	22.11.2013 13:46:47	Test Case	manually created
New Entry	Testmanager	12.09.2012 14:21:01	Test Case	manually created
New Entry3	Testmanager	12.09.2012 14:22:11	Test Suite	manually created
Regressiontest	Testmanager	22.11.2013 13:46:46	Test Case	manually created
Sample	Testmanager	22.11.2013 13:46:42	Test Case	manually created
Treiber	Testmanager	22.11.2013 13:46:41	Test Case	manually created

Figure 327 – Copy of UDA

Rename the attribute by changing the name of the attribute (see Figure 328). If the checkbox **Copy values** is clicked, the values of the attribute to copy (example: values of "Milestone") are copied too. If the checkbox is not clicked, only the attribute is copied (not the values).



Treiber

Created: 09.12.2013 17:13:31

Owner: Testmanager

Type: Attribute ☑ Copy values

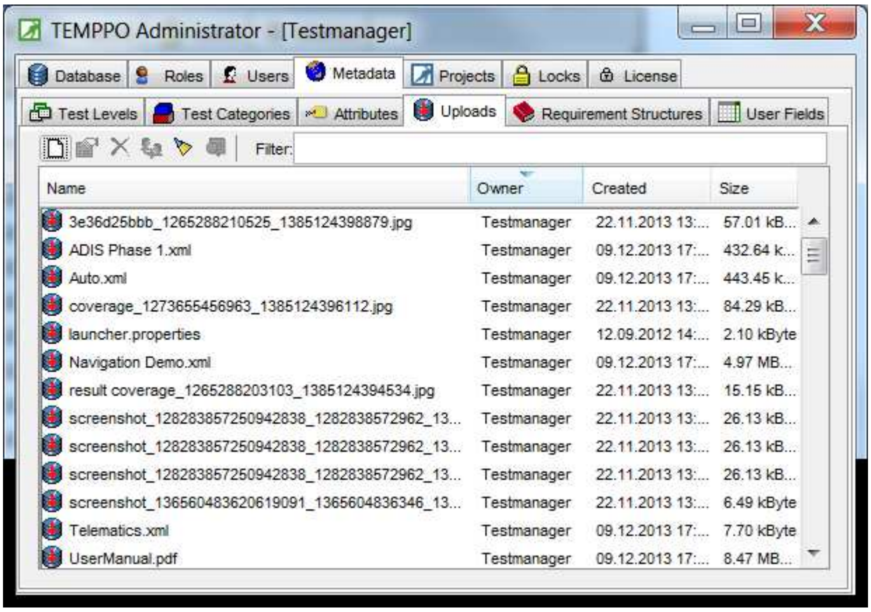
Description:

Figure 328 – Copy and Rename

5.4.5 Uploads

If you activate the tab **Uploads** (Figure 329), you can

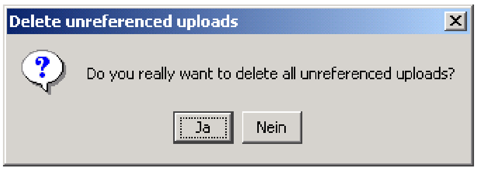
- **Upload new files without any reference**
- **View uploads properties**
- **Delete uploads from the database**
- **Delete all non-referenced uploads**



Name	Owner	Created	Size
3e36d25bbb_1265288210525_1385124398879.jpg	Testmanager	22.11.2013 13:...	57.01 kB...
ADIS Phase 1.xml	Testmanager	09.12.2013 17:...	432.64 k...
Auto.xml	Testmanager	09.12.2013 17:...	443.45 k...
coverage_1273655456963_1385124396112.jpg	Testmanager	22.11.2013 13:...	84.29 kB...
launcher.properties	Testmanager	12.09.2012 14:...	2.10 kByte
Navigation Demo.xml	Testmanager	09.12.2013 17:...	4.97 MB...
result coverage_1265288203103_1385124394534.jpg	Testmanager	22.11.2013 13:...	15.15 kB...
screenshot_128283857250942838_1282838572962_13...	Testmanager	22.11.2013 13:...	26.13 kB...
screenshot_128283857250942838_1282838572962_13...	Testmanager	22.11.2013 13:...	26.13 kB...
screenshot_128283857250942838_1282838572962_13...	Testmanager	22.11.2013 13:...	26.13 kB...
screenshot_136560483620619091_1365604836346_13...	Testmanager	22.11.2013 13:...	6.49 kByte
Telematics.xml	Testmanager	09.12.2013 17:...	7.70 kByte
UserManual.pdf	Testmanager	09.12.2013 17:...	8.47 MB...

Figure 329 – Metadata: Uploads

If you activate **Delete all non-referenced uploads**, a confirmation window (see Figure 330) is displayed, before deleting all non-referenced uploads.



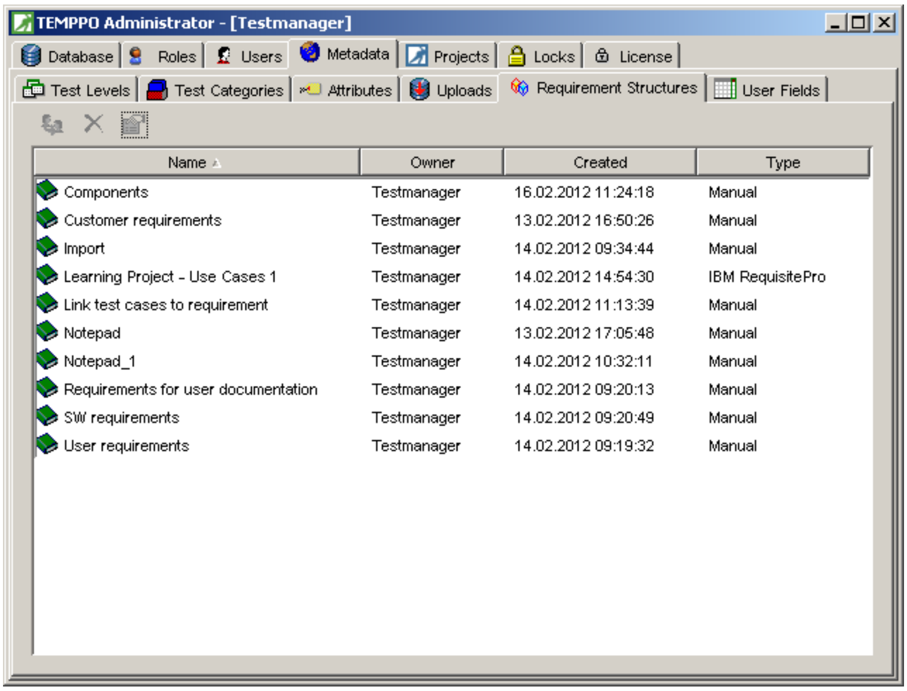
Do you really want to delete all unreferenced uploads?

Figure 330 – Confirmation window

5.4.6 Requirement Structures

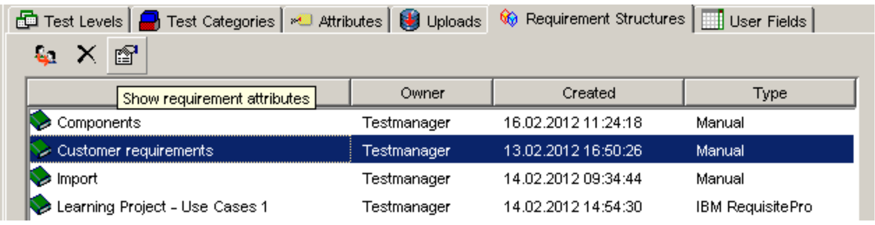
With requirement structures you can assign requirements to test cases or test packages. In the list you see all existing requirement structures (see Figure 331).

You can create new requirement structures with TEMPPO Requirement Manager.



Name	Owner	Created	Type
Components	Testmanager	16.02.2012 11:24:18	Manual
Customer requirements	Testmanager	13.02.2012 16:50:26	Manual
Import	Testmanager	14.02.2012 09:34:44	Manual
Learning Project - Use Cases 1	Testmanager	14.02.2012 14:54:30	IBM RequisitePro
Link test cases to requirement	Testmanager	14.02.2012 11:13:39	Manual
Notepad	Testmanager	13.02.2012 17:05:48	Manual
Notepad_1	Testmanager	14.02.2012 10:32:11	Manual
Requirements for user documentation	Testmanager	14.02.2012 09:20:13	Manual
SW requirements	Testmanager	14.02.2012 09:20:49	Manual
User requirements	Testmanager	14.02.2012 09:19:32	Manual

Figure 331 – Metadata: Requirement Structures



	Owner	Created	Type
Components	Testmanager	16.02.2012 11:24:18	Manual
Customer requirements	Testmanager	13.02.2012 16:50:26	Manual
Import	Testmanager	14.02.2012 09:34:44	Manual
Learning Project - Use Cases 1	Testmanager	14.02.2012 14:54:30	IBM RequisitePro

Figure 332 – Show requirement attributes

With the button **Show requirement attributes** the assigned attributes are shown in a new window (see Figure 333). It’s only possible to un/assign attributes, when the requirement structure is *not* imported.

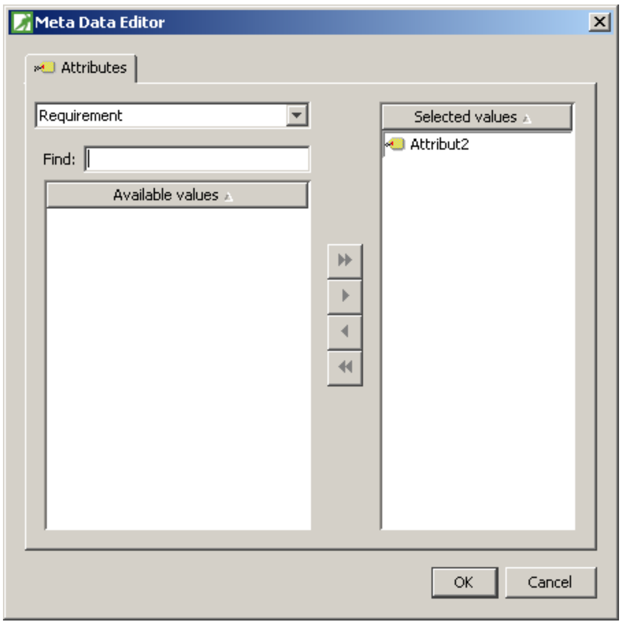
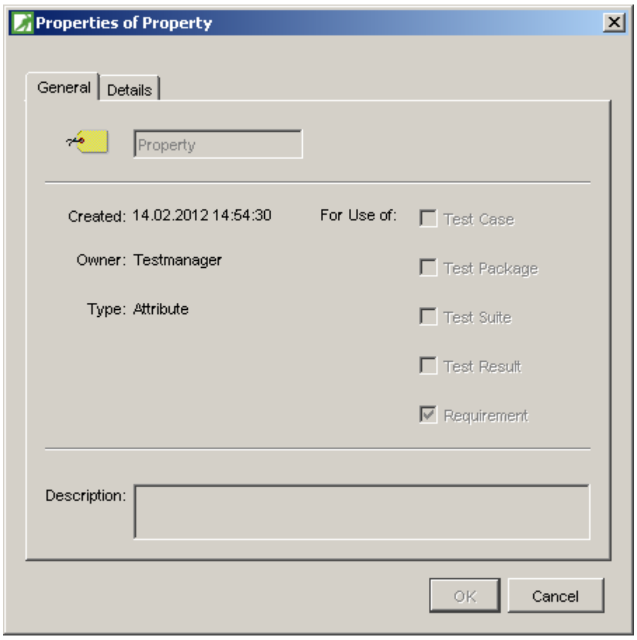


Figure 333 – Requirement attributes

With a double click on an attribute (tab **attributes**) the properties of the attribute are shown (see Figure 334). It’s not possible to change the requirement attribute, because it’s imported from DOORS or RequisitePro.



Property

Created: 14.02.2012 14:54:30

Owner: Testmanager

Type: Attribute

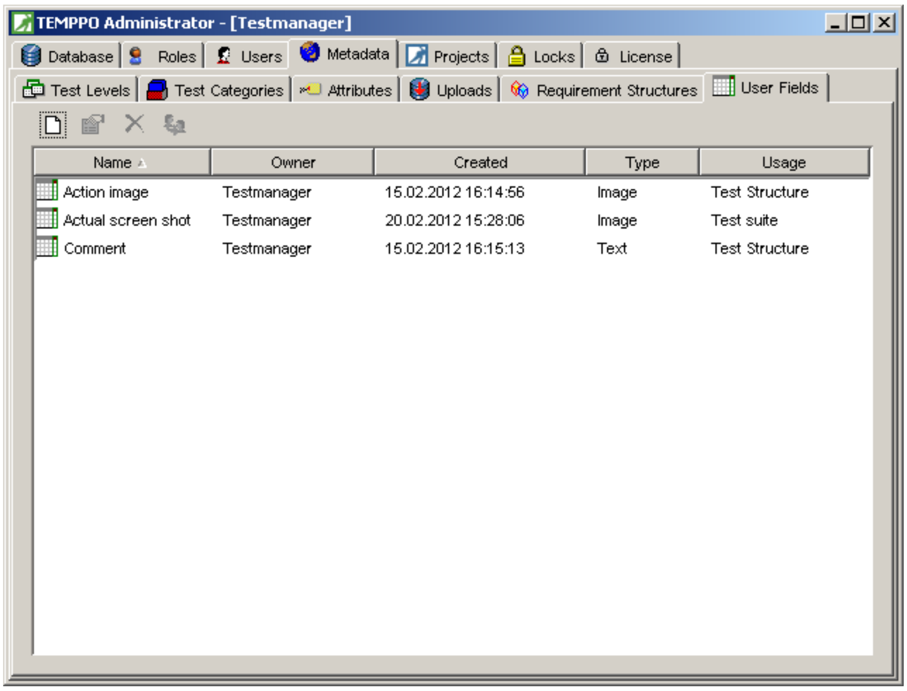
For Use of:

□ Test Case
□ Test Package
□ Test Suite
□ Test Result
☑ Requirement

Description:

Figure 334 – Properties of attribute

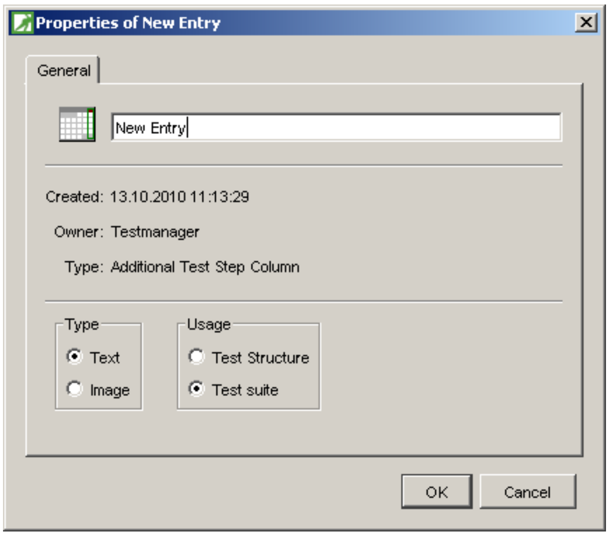
5.4.7 User Fields



Name	Owner	Created	Type	Usage
Action image	Testmanager	15.02.2012 16:14:56	Image	Test Structure
Actual screen shot	Testmanager	20.02.2012 15:28:06	Image	Test suite
Comment	Testmanager	15.02.2012 16:15:13	Text	Test Structure

Figure 335 – Metadata: User Fields

For test steps new columns can be defined for test structures and test suites. When pressing the button **New** (see Figure 336), you can select between a text and image column for test structure or suite.



New Entry

Created: 13.10.2010 11:13:29

Owner: Testmanager

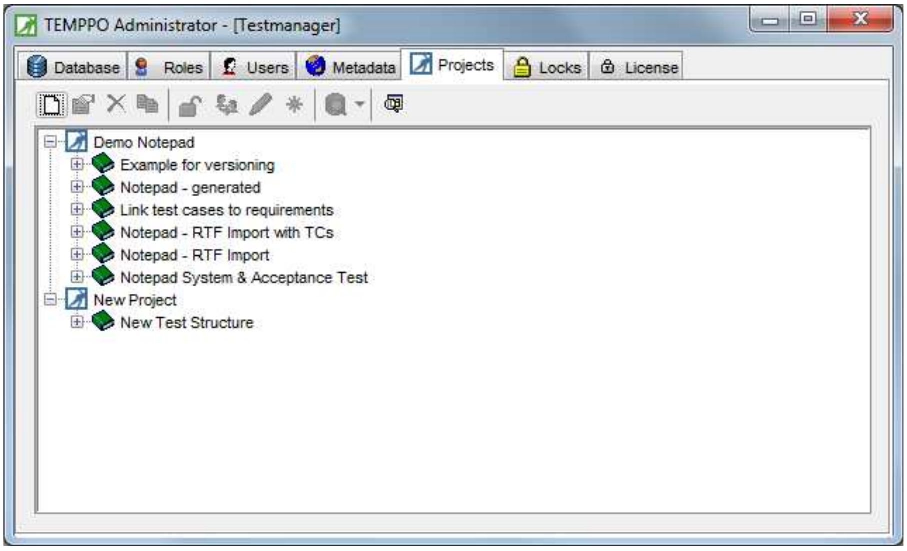
Type: Additional Test Step Column

Type
● Text
○ Image

Usage
○ Test Structure
● Test suite

Figure 336 – Additional test step column

5.5 Projects



- Demo Notepad
  - Example for versioning
  - Notepad - generated
  - Link test cases to requirements
  - Notepad - RTF Import with TCs
  - Notepad - RTF Import
  - Notepad System & Acceptance Test
- New Project
  - New Test Structure

Figure 337 – Administration of Projects

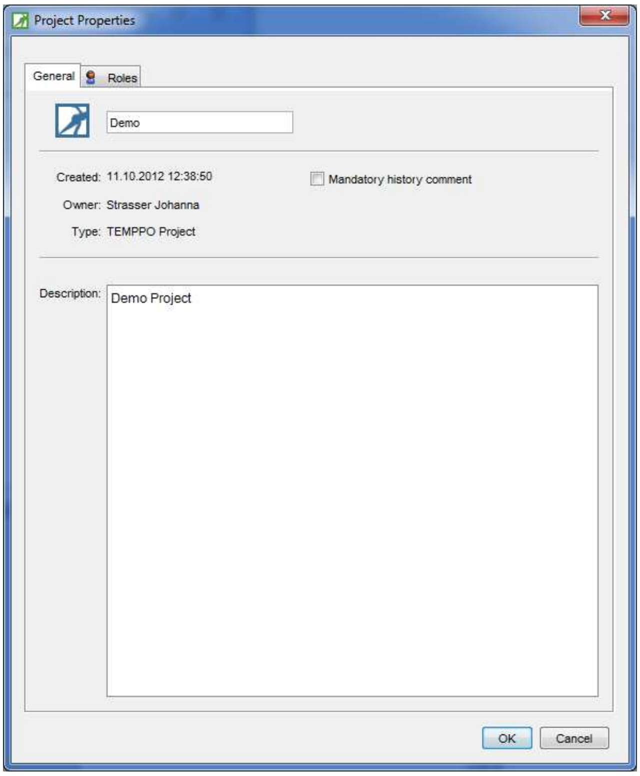
The **Project** tab is only active, if the user has a role with sufficient rights. A TEMPPO Superuser has access to all projects.

If allowed due to role privileges, you can **create and copy projects, delete test suites, labels, branches, test structures or even entire projects and assign metadata to projects**.

5.5.1 Creating a project

A new project can only be created in TEMPPO Administrator.

After pressing the button **New**, Figure 338 is displayed. The user specifies a unique name and a description. TEMPPO automatically supplies the creation date and owner, which is taken from the Windows account.



Demo

Created: 11.10.2012 12:38:50

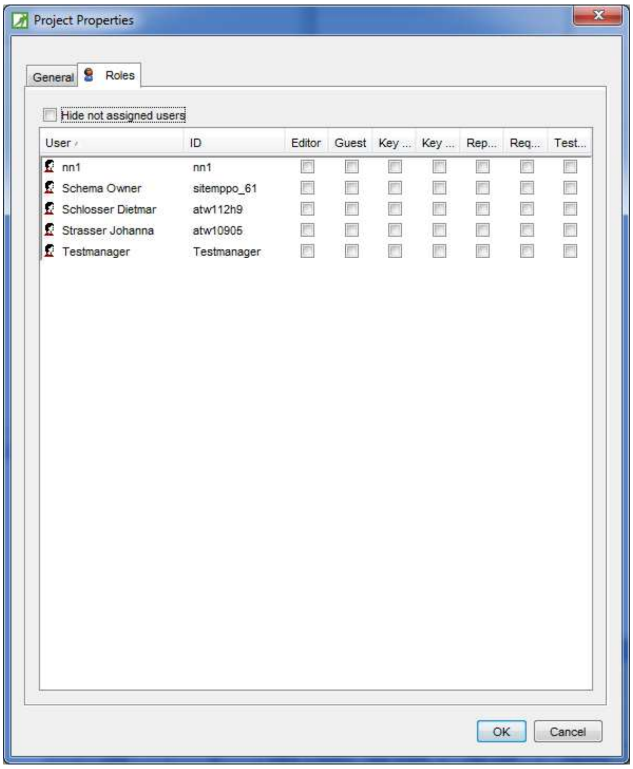
Owner: Strasser Johanna

Type: TEMPPO Project

□ Mandatory history comment

Description: Demo Project

Figure 338 – Project Properties – New Project



□ Hide not assigned users

User	ID	Editor	Guest	Key ...	Key ...	Rep...	Req...	Test...
nn1	nn1	□	□	□	□	□	□	□
Schema Owner	sitemppo_61	□	□	□	□	□	□	□
Schlosser Dietmar	atw112h9	□	□	□	□	□	□	□
Strasser Johanna	atw10905	□	□	□	□	□	□	□
Testmanager	Testmanager	□	□	□	□	□	□	□

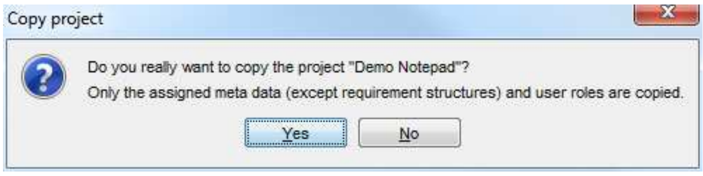
Figure 339 – Project Properties: Roles

In the tab **Roles**, roles can be assigned to users (see Figure 339). By pressing **OK** the project is created. Then the project can be edited and worked on in TEMPPO.

5.5.2 Copying a project

A new project can only be copied in TEMPPO Administrator.

If a project selected and the button **Copy current project** is pressed, Figure 338 is displayed.



Copy project

Do you really want to copy the project "Demo Notepad"?
Only the assigned meta data (except requirement structures) and user roles are copied.

Figure 340 - Copy project confirmation

The user is asked whether he wants to copy the current selected project including all assigned **meta data** except requirement structures. If the user presses **Yes**, the project including assigned users plus meta data except requirement structures is copied.

5.5.3 Deletions

The delete button is active, if a node (project, test structure (version), test suite) in the tree is selected. By clicking it, the corresponding object is deleted after you confirmed the verification message.

ⓘ **You cannot delete the main latest label or any latest label in a branch!**

5.5.4 Test Suites: check consistency, unlock

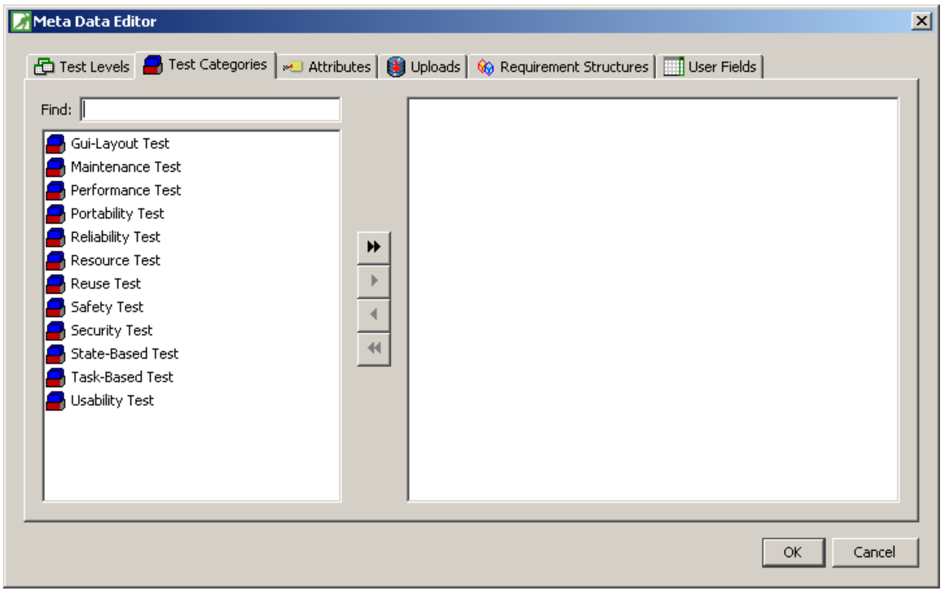
The buttons **Check Consistency** of a selected test suite and **Unlock** selected test suite are active, if a test suite is selected.

Furthermore it is possible to unlock a whole test suite/structure. In case of problems during exporting and importing test suites it may happen that references remain in the central database. To solve these problems it is necessary to unlock the test suite. Figure 337 shows the locked test suite **Test Suite for export**. The **Lock** button becomes enabled when the test suite is selected.

ⓘ **If you unlock a test suite/structure no more imports are possible. After that, exported test suites are of no more use and can be deleted from the Export Database.**

5.5.5 Assign Metadata

For a chosen project all kinds of metadata (test level, test category, attributes, uploads, requirement structures, user fields) can be assigned to the project. Metadata is defined project spanning. The assignment is made per project.



- Gui-Layout Test
- Maintenance Test
- Performance Test
- Portability Test
- Reliability Test
- Resource Test
- Reuse Test
- Safety Test
- Security Test
- State-Based Test
- Task-Based Test
- Usability Test

Figure 341 – Metadata Editor

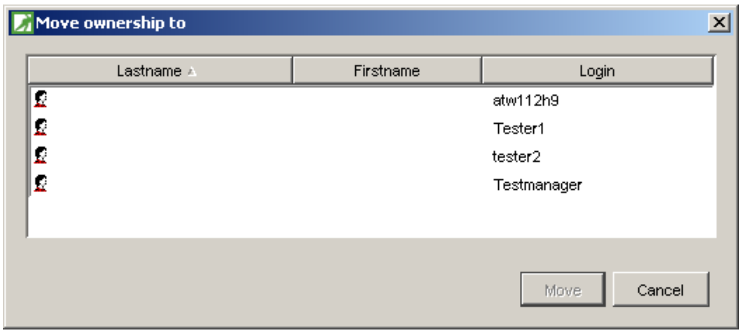
Example: On the left side all available test categories are listed. By clicking the buttons >>, >, <,<< the test categories can be moved to right list box. The assignments on the right are saved by pressing the button "Apply".

5.5.6 Move Ownership

In the TEMPPO Administrator there is the possibility to move the ownership of the whole project or of parts of it. A part of a project is a test structure or a test suite.

The button **Move Ownership** in the project tab (see Figure 337) is enabled, if a project (or a test structure or a test suite) is selected.

After pressing the button "Move Ownership" the following dialog comes up.



Lastname	Firstname	Login
		atw112h9
		Tester1
		tester2
		Testmanager

Figure 342 – Move Ownership

The new owner of the project (or part of it) can be chosen. By clicking "ok" and a project was selected, the owner of all items of the project (test structure, test package, test case, test suite) that have the same owner as the project and that are not locked by other users, is changed to the new selected one.

If a test structure was selected and the move ownership button is pressed, the test structure, the test packages and test cases with the same owner as the test structure are changed to the newly selected one.

If a test suite was selected, the owner of the test suite is changed.

5.5.7 Freeze, Unfreeze

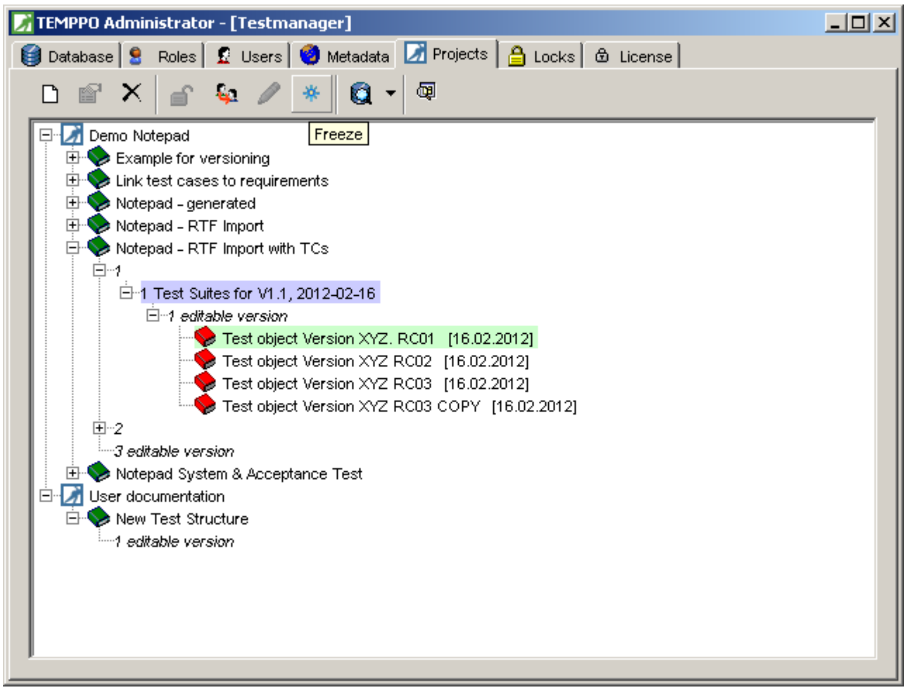

TEMPPO Administrator - [Testmanager]
Database
Roles
Users
Metadata
Projects
Locks
License
Freeze
Demo Notepad
Example for versioning
Link test cases to requirements
Notepad - generated
Notepad - RTF Import
Notepad - RTF Import with TCs
1
1 Test Suites for V1.1, 2012-02-16
1 editable version
Test object Version XYZ. RC01 [16.02.2012]
Test object Version XYZ RC02 [16.02.2012]
Test object Version XYZ RC03 [16.02.2012]
Test object Version XYZ RC03 COPY [16.02.2012]
2
3 editable version
Notepad System & Acceptance Test
User documentation
New Test Structure
1 editable version

Figure 343 – Freeze, Unfreeze

From the development process point of view it may be necessary for a test manager to “freeze” a test structure or a test suite for the purpose of providing and keeping it read-only.

In general, test structures and test suites can be frozen.

A frozen test structure cannot be edited any more. It can be opened in read-only mode. A freeze- action on a non-latest version means that it is not possible to create any branches.

If a test suite is selected and the freeze button is pressed, this test suite is frozen which means that the test suite is not editable any more. No results can be added or changed. The test suite can be opened in read-only mode. The precondition of freezing a test suite is that the corresponding test structure version is frozen.

Every frozen test suite can be unfrozen explicitly. This means that the frozen test suite is selected and the unfreeze button is clicked. After the unfreeze action is done, the test suite is editable again.

Every frozen test suite can be unfrozen implicitly too. This means that the test structure version has to be unfrozen first.

If the user pressed the unfreeze button for a test structure version, he is informed if there are frozen test suites and that these frozen test suites (and the test structure version) will be unfrozen, too. The user can unfreeze all or cancel the action.

If a frozen test structure or test suite is selected, the ‘freeze’ button changes its text in icon and bubble help to “unfreeze”. By pressing the button the test structure or test suite can be made editable. The default TEMPPO Superuser and the Key User role get the rights to freeze / unfreeze test structure versions and/or test suites.

Frozen test structures can be exported. Frozen test suites cannot be exported. No import to frozen test structures/test suites is possible. A new test suite can only be generated from a frozen test structure in any branch version, but not in the main version. If a main version is frozen and activated, a new test suite cannot be generated, the new test suite – menu item is not enabled.

If a branch version or latest version is frozen and activated, a new test suite can be generated and the new test suite – menu item is enabled.

If a version is checked in and is frozen, no branch can be generated and therefore no new test suite can be generated.

The frozen test structure/test suite cannot be edited. There is only the possibility to generate analysis or reports and filters.

5.6 Locks

This tab is only active, if a user with corresponding rights is connected to the database In TEMPPO Administrator it is also possible to find locks. When activating the tab **Locks** in the left part of the windows the projects together with test structure versions and test suites are displayed.

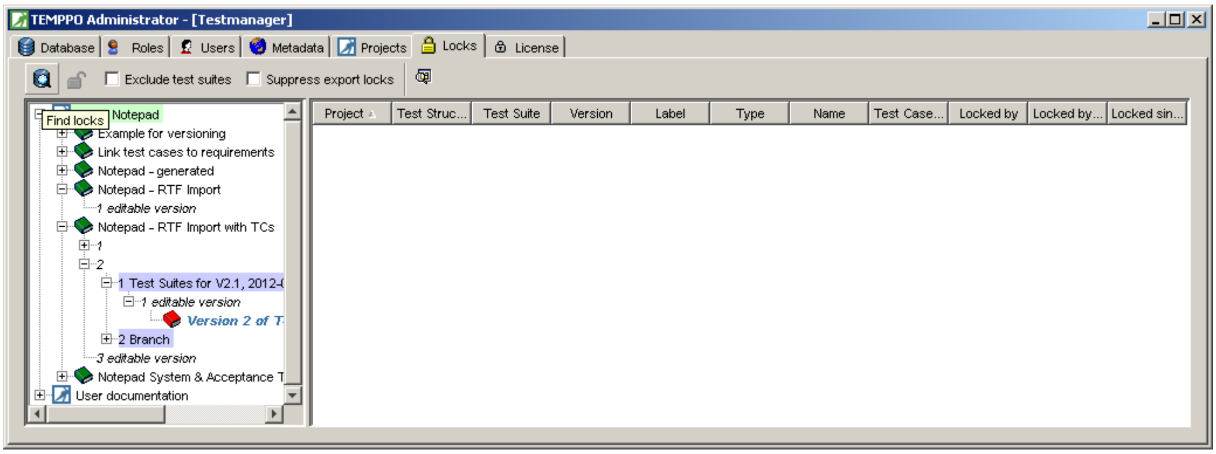
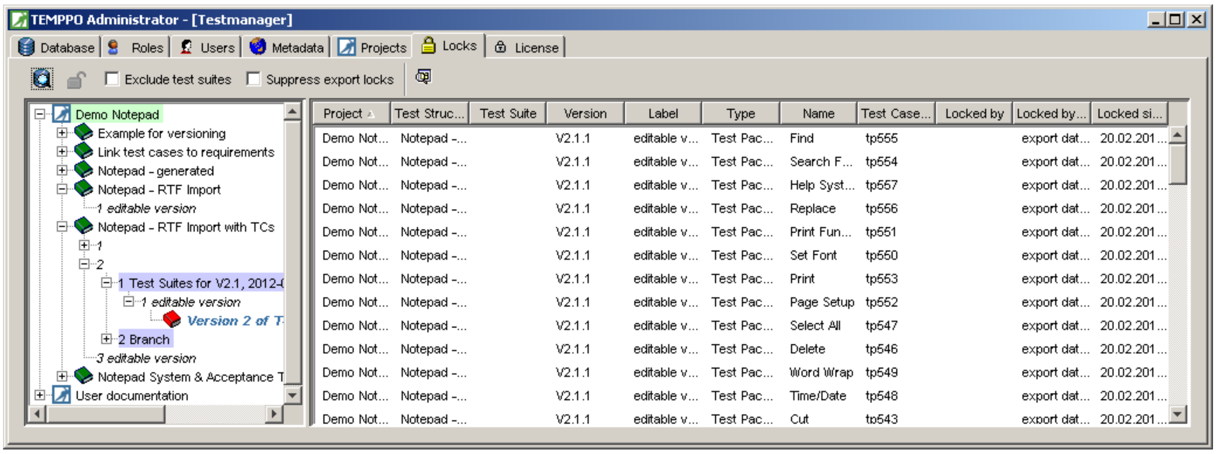


Figure 344 – TEMPPO Administrator – Locks

If you select any item of the tree and press the button **Find locks**, the corresponding locks are displayed in the table. Depending on the selected item (and of course the available locks) the table will be quite large.



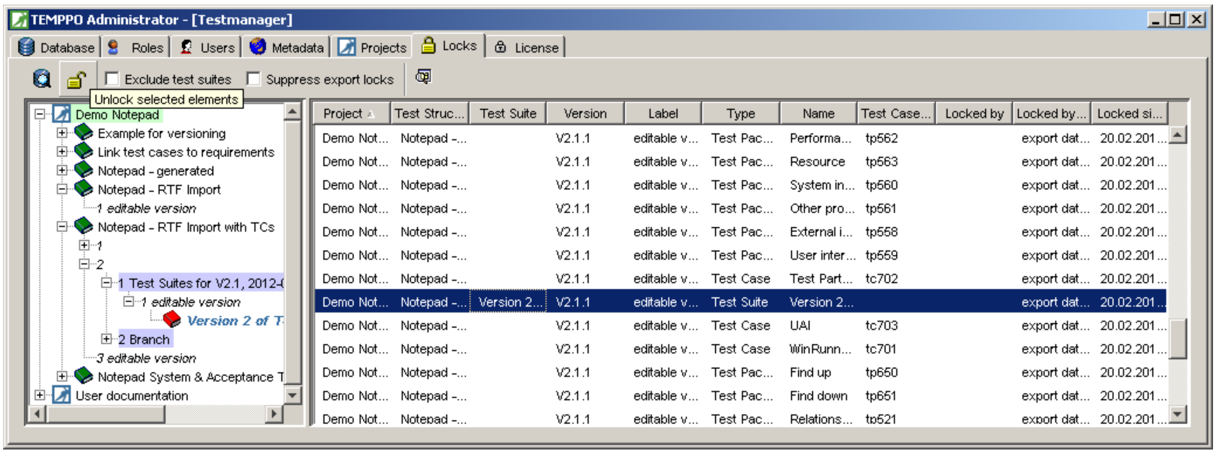
Project	Test Struc...	Test Suite	Version	Label	Type	Name	Test Case...	Locked by	Locked by...	Locked si...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Find	tp555		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Search F...	tp554		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Help Syst...	tp557		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Replace	tp556		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Print Fun...	tp551		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Set Font	tp550		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Print	tp553		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Page Setup	tp552		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Select All	tp547		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Delete	tp546		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Word Wrap	tp549		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Time/Date	tp548		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Cut	tp543		export dat...	20.02.201...

Figure 345 – TEMPPO Administrator – Large lock table

Additionally you have the possibility to start a more concrete find by activating the checkboxes **Exclude test suites** and **Suppress export locks**.

- Exclude test suites:
  If you want to see the locks of a project without test suites, activate that checkbox
- Suppress export locks:
  If a test suite is exported to XML or DB, each item is locked. This may cause an large amount of locked item. You hide those items by activating that checkbox.

Locked items can be unlocked by selecting any them and pressing the button **Unlock selected items**.

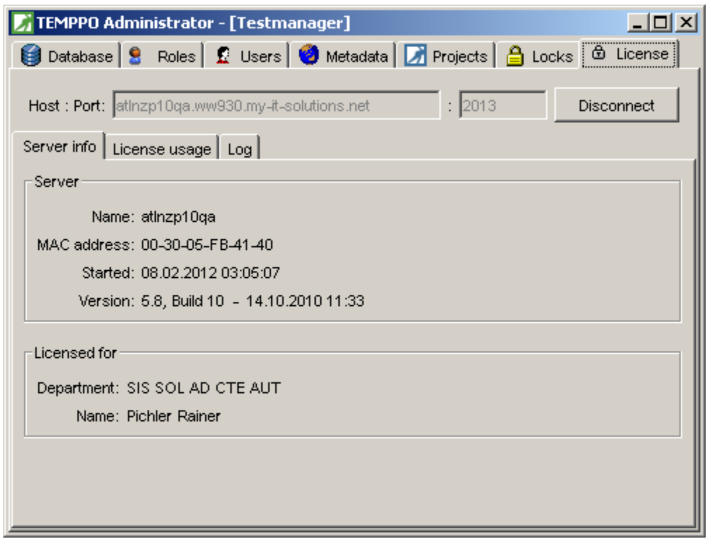


☐ Exclude test suites ☐ Suppress export locks

Project	Test Struc...	Test Suite	Version	Label	Type	Name	Test Case...	Locked by	Locked by...	Locked si...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Performa...	tp562		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Resource	tp563		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	System in...	tp560		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Other pro...	tp561		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	External i...	tp558		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	User inter...	tp559		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Case	Test Part...	tc702		export dat...	20.02.201...
Demo Not...	Notepad -...	Version 2...	V2.1.1	editable v...	Test Suite	Version 2...			export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Case	UAI	tc703		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Case	WinRunn...	tc701		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Find up	tp650		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Find down	tp651		export dat...	20.02.201...
Demo Not...	Notepad -...		V2.1.1	editable v...	Test Pac...	Relations...	tp521		export dat...	20.02.201...

Figure 346 – Unlock

5.7 License Server Connection



Host : Port: atlnzp10qa.ww930.my-it-solutions.net : 2013

Server

Name: atlnzp10qa
MAC address: 00-30-05-FB-41-40
Started: 08.02.2012 03:05:07
Version: 5.8, Build 10 - 14.10.2010 11:33

Licensed for

Department: SIS SOL AD CTE AUT
Name: Pichler Rainer

Figure 347: License server connection

The licensing mechanism allows two ways of using TEMPPO:

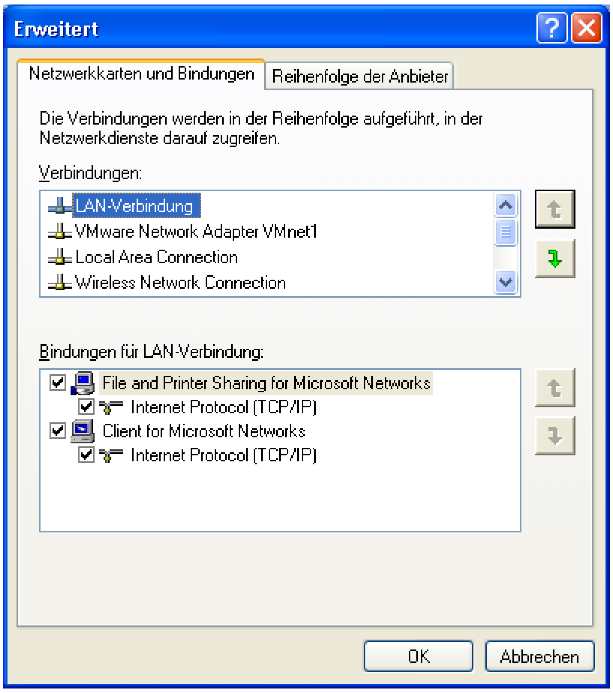
- local license:a valid license file (license.dat) must be located in the TEMPPO installation directory (e.g. c:\programme\temppo 5.4).
- license server: an existing license server is specified in this tab (for installing a license server see chapter JLMS Installation in the TEMPPO Installation Guide). You must specify the name of the machine (or IP

address) and the port number (usually 2000) where the license server is running. When connected to it, you can see information about it in the sub-tabs Server info, License usage, Log.

5.7.1 Using a license server on a computer with several network interface cards

When working on a computer with several NICs (network interface cards) the connection to the license server might cause problems, because the wrong IP address is sent to the license server. This problem can be solved in two ways:

1. Reorder your network cards:
   - Open **Control Panel – Network Connections**
   - Menu **Extended – Extended Settings…**
   - The NIC connected to LAN must be the first in the list (see screenshot)



Die Verbindungen werden in der Reihenfolge aufgeführt, in der Netzwerkdienste darauf zugreifen.

Verbindungen:

- LAN-Verbindung
- VMware Network Adapter VMnet1
- Local Area Connection
- Wireless Network Connection

Bindungen für LAN-Verbindung:

- ☑ File and Printer Sharing for Microsoft Networks
  - ☑ Internet Protocol (TCP/IP)
- ☑ Client for Microsoft Networks
  - ☑ Internet Protocol (TCP/IP)

Figure 348: Order of network interface cards

2. Add TEMPPO start option:
   - Open `<Installation Directory>\launcher.properties` with a text editor
   - Add `-Djava.rmi.hostname=<IP address>` to `TEMPPO.vmargs`, e.g.

```
TEMPPO.vmargs = -Xms128m -Xmx256m -DentityExpansionLimit=1000000 -
Djava.rmi.hostname=158.226.235.13
```

6 JUnit

Since TEMPPO V3.0, it is possible to manage and execute JUnit test methods with TEMPPO. The following paragraphs describe the scope of the JUnit extension.

6.1 Terms

Testing terms are used differently in the JUnit community and in TEMPPO. In order to clarify the meanings, the following table serves as a reference.

TEMPPO	Description	JUnit	Defined in JUnit by
Test structure	The entire tree of test packages and test cases	-	-
Test package	A hierarchy of test packages and test cases	test suite (not exactly the same meaning)	a) *suite()* method in an arbitrary class b) the list of all test methods in a sub-class of *TestCase*
test suite	A subset of test cases from the test structure, intended for execution in a certain test. Contains recorded test results.	Test suite (not exactly the same meaning)	as above
test case	A set of test inputs, execution conditions, and expected results developed for a particular objective.	Test method	a method in a sub-class of *TestCase.* Naming convention: starts with “test”.
Passed	Test result, indicating that the actual output matches the expected.	-	-
Failed	Test result, indicating that the actual output does not match the expected.	Failure	return code of test method invocation.
Blocked	Test result, indicating that an error or exception occurred during test execution; no comparison of actual and expected output was possible.	Error	return code of test method invocation.

Table 3 – Terms used in TEMPPO and JUnit

6.2 Configuration

Two configurations are necessary to use the JUnit extension: *test collector* and *classpath.* The configuration is done in the TEMPPO application via menu **Windows > Options**, once for the whole TEMPPO session. The *classpath* is entered in the "Arguments" field in the usual format:

-cp path1;path2;...

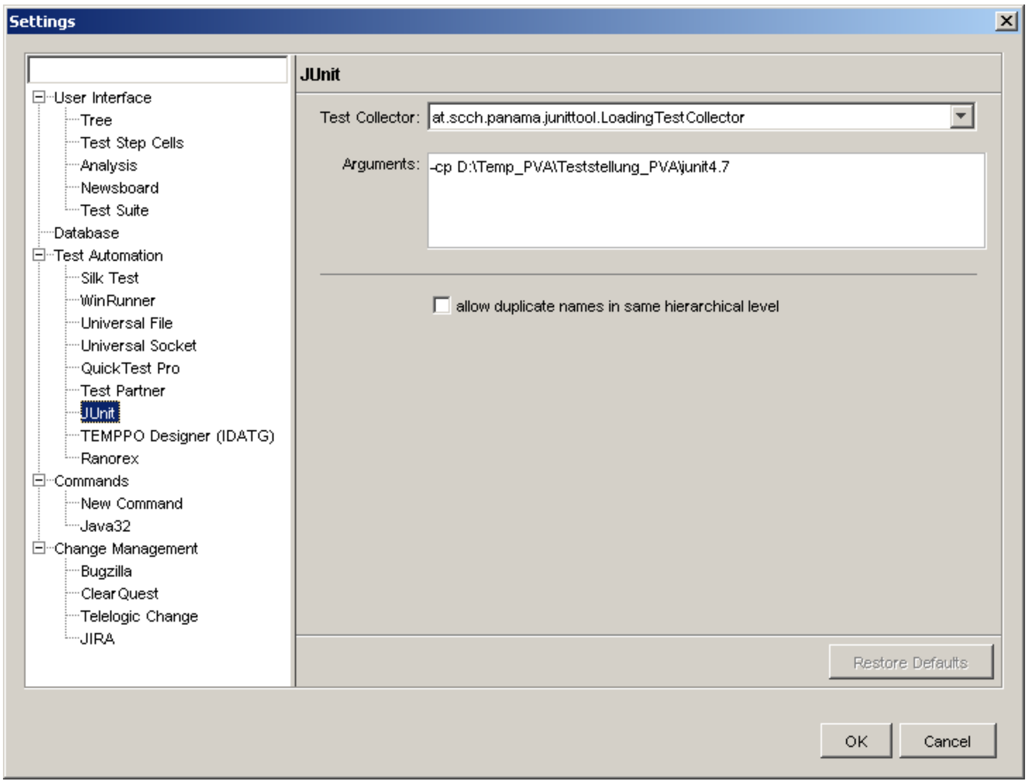
As in the JUnit framework, it must point to both the JUnit tests and the Java classes under test. The *test collector* is the part of the JUnit extension, which searches for JUnit test cases in the specified *classpath*. Following test collectors are available:

- SimpleTestCollector: searches for JUnit classes in files matching the pattern

```
classFileName.endsWith(".class") &&
classFileName.indexOf('$') < 0 &&
classFileName.indexOf("Test") > 0
```

- LoadingTestCollector: searches for JUnit classes by instantiating all *.class files in the classpath and verifying if:

```
class.containsSuiteMethod() ||
(
 class.extends(junit.framework.Test) &&
 class.isPublic() &&
 class.hasPublicConstructor()
)
```



JUnit

Test Collector: at.scch.panama.junittool.LoadingTestCollector

Arguments: -cp D:\Temp_PVA\Teststellung_PVA\junit4.7

☐ allow duplicate names in same hierarchical level

Figure 349 – Settings for import and execution of JUnit test cases

6.3 Import JUnit test cases

6.3.1 Description and steps

The import function reads JUnit test cases as well as their suite structure from a JUnit test class. It creates a new test package in TEMPPO, which contains further test packages and eventually test cases, according to the suite structure found in the JUnit test class. The generated test cases refer to JUnit test methods.

You can access the import function through the TEMPPO menu *Tools>JUnit>Import...* This menu is activated only when an item in the test structure is selected that can host a new test package, i.e. a test package or the root node of the test structure.

A wizard guides you through the steps necessary to import the JUnit test cases. The following steps are encompassed:

1. *Introduction*: Explains the steps of the import procedure

Test Class Selection: Provides the list of JUnit test classes you may import. Only classes found in the *classpath* are listed here (see 6.2).

Test Case Selection: Provides the hierarchical structure of test cases in the suite defined by the selected JUnit class. Select those test cases that you want to import into TEMPPO.

Package Default Values: Allows to set default values for all new test packages

Test Case Default Values: Allows setting default values for all new test cases.

Summary: Shows a list of your choices from previous steps.

Here is a sample import sequence.

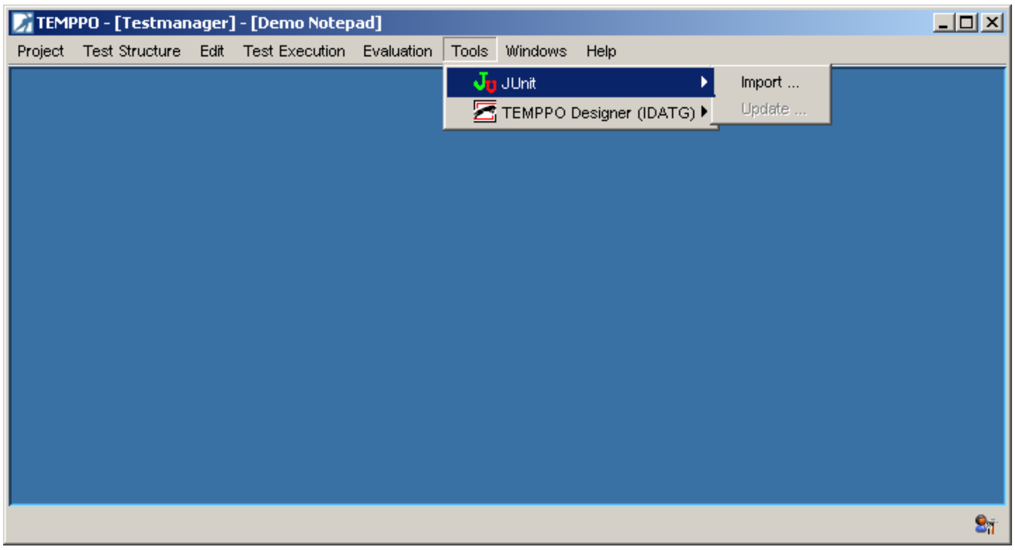
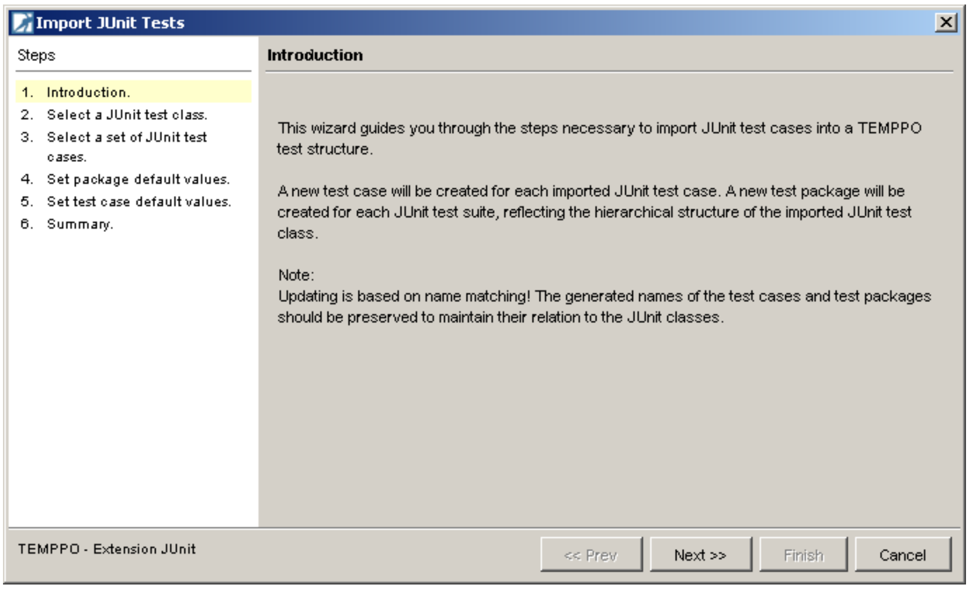


Figure 350 – Starting the import procedure



1. Introduction.
2. Select a JUnit test class.
3. Select a set of JUnit test cases.
4. Set package default values.
5. Set test case default values.
6. Summary.

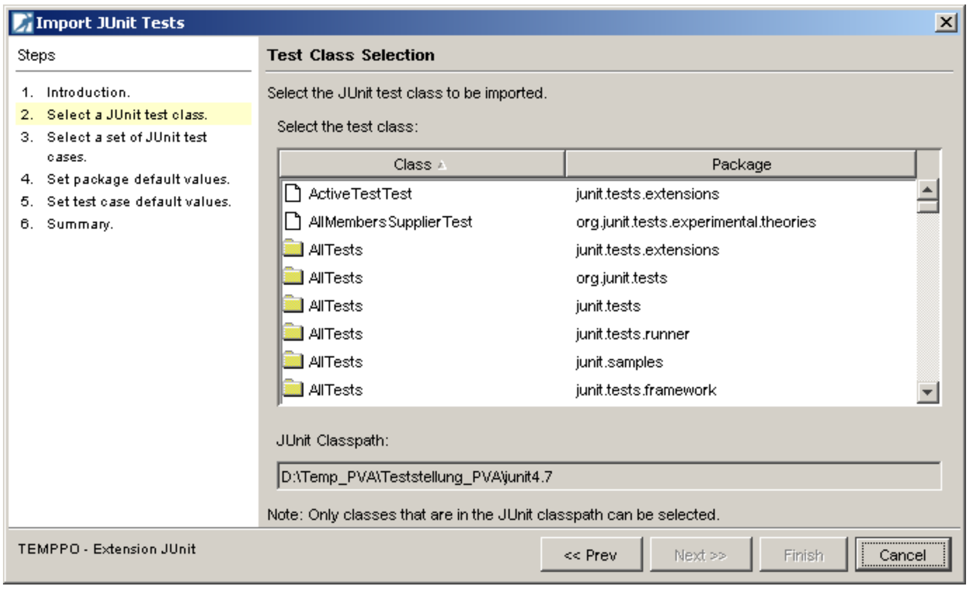
This wizard guides you through the steps necessary to import JUnit test cases into a TEMPPO test structure.

A new test case will be created for each imported JUnit test case. A new test package will be created for each JUnit test suite, reflecting the hierarchical structure of the imported JUnit test class.

Note:
Updating is based on name matching! The generated names of the test cases and test packages should be preserved to maintain their relation to the JUnit classes.

Figure 351 – General description of the following steps.

In the next step, the JUnit test class can be selected. All classes with names that start with “All Tests” are shown with a folder icon, all other classes get a sheet icon. This is in conformance with the TestRunner of the original JUnit distribution. The icons are to some extent misleading because they suggest that only test classes with folder icons will be converted to TEMPPO test packages. However, each class that can be selected in this step will be converted to a (potentially hierarchically structured) test package.



Import JUnit Tests

Steps

1. Introduction.
2. Select a JUnit test class.
3. Select a set of JUnit test cases.
4. Set package default values.
5. Set test case default values.
6. Summary.

Test Class Selection

Select the JUnit test class to be imported.

Select the test class:

Class	Package
ActiveTestTest	junit.tests.extensions
AllMembersSupplierTest	org.junit.tests.experimental.theories
AllTests	junit.tests.extensions
AllTests	org.junit.tests
AllTests	junit.tests
AllTests	junit.tests.runner
AllTests	junit.samples
AllTests	junit.tests.framework

JUnit Classpath:

D:\Temp_PVA\Teststellung_PVA\junit4.7

Note: Only classes that are in the JUnit classpath can be selected.

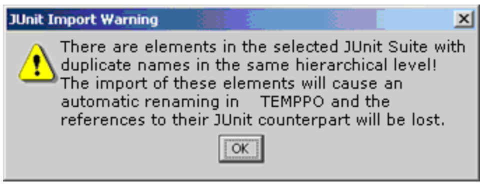
TEMPPO - Extension JUnit

Figure 352 – Step 2: A JUnit test class can be selected, by which a JUnit test suite is defined.

After clicking *next*, the *test collector* specified in the Administrator (see 6.2) starts searching for JUnit test classes. JUnit allows having nested test suites (one test suite containing others) and it also allows naming two or more test suites having the same parent test suite identically, e.g.

```
All Junit Tests
junit.samples.VectorTest
junit.samples.money.MoneyTest
Framework Tests
        Framework Tests
        Framework Tests
        Framework Tests
```

Since TEMPPO will create a Test Package for each JUnit test suite and TEMPPO does not allow identical names of items with the same parent, a warning message will be raised, if the test collector finds test suites named like described above (for further details see 6.3.2).

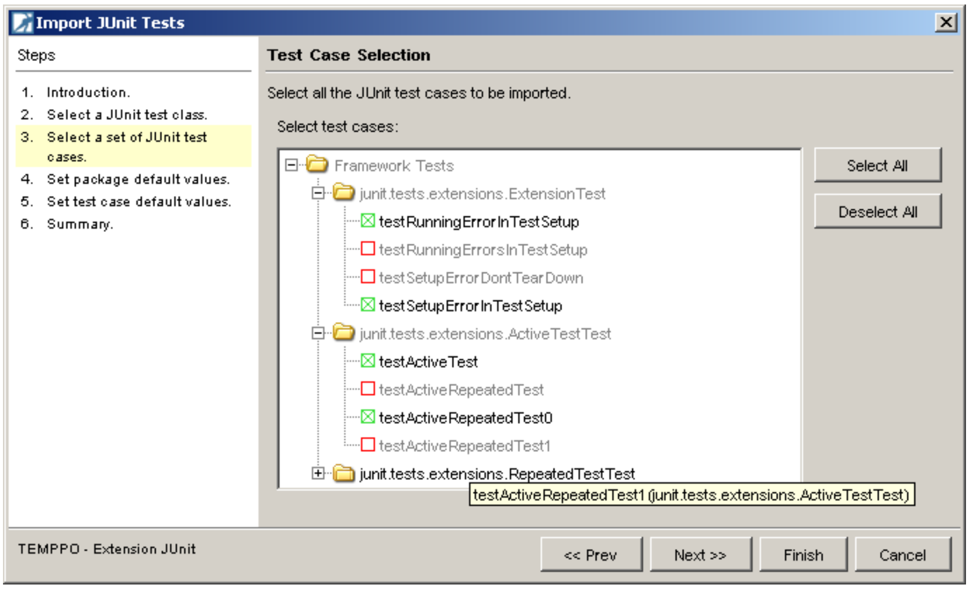


JUnit Import Warning

There are elements in the selected JUnit Suite with duplicate names in the same hierarchical level!
The import of these elements will cause an automatic renaming in TEMPPO and the references to their JUnit counterpart will be lost.

Figure 353: Warning of duplicate test suite names

The related test suites are marked accordingly and can’t be selected for importing.



Import JUnit Tests

Steps

1. Introduction.
2. Select a JUnit test class.
3. Select a set of JUnit test cases.
4. Set package default values.
5. Set test case default values.
6. Summary.

Test Case Selection

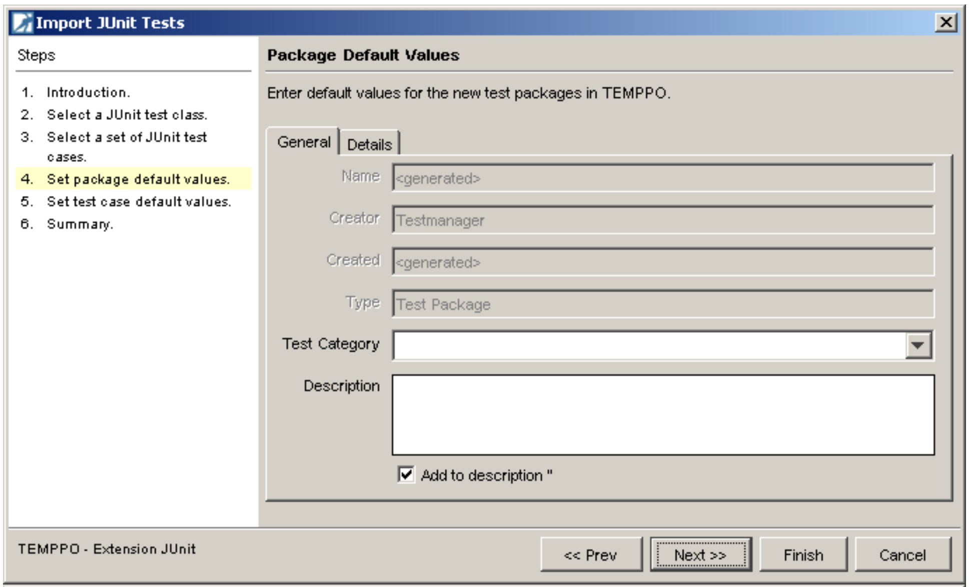
Select all the JUnit test cases to be imported.

Select test cases:

- Framework Tests
  - junit.tests.extensions.ExtensionTest
    - ☒ testRunningErrorInTestSetup
    - □ testRunningErrorsInTestSetup
    - □ testSetupErrorDontTearDown
    - ☒ testSetupErrorInTestSetup
  - junit.tests.extensions.ActiveTestTest
    - ☒ testActiveTest
    - □ testActiveRepeatedTest
    - ☒ testActiveRepeatedTest0
    - □ testActiveRepeatedTest1
  - junit.tests.extensions.RepeatedTestTest

testActiveRepeatedTest1 (junit.tests.extensions.ActiveTestTest)

Figure 354 – Step 3: All or a subset of JUnit test cases can be selected.



Import JUnit Tests

Steps

1. Introduction.
2. Select a JUnit test class.
3. Select a set of JUnit test cases.
4. Set package default values.
5. Set test case default values.
6. Summary.

Package Default Values

Enter default values for the new test packages in TEMPPO.

General | Details

Name: <generated>

Creator: Testmanager

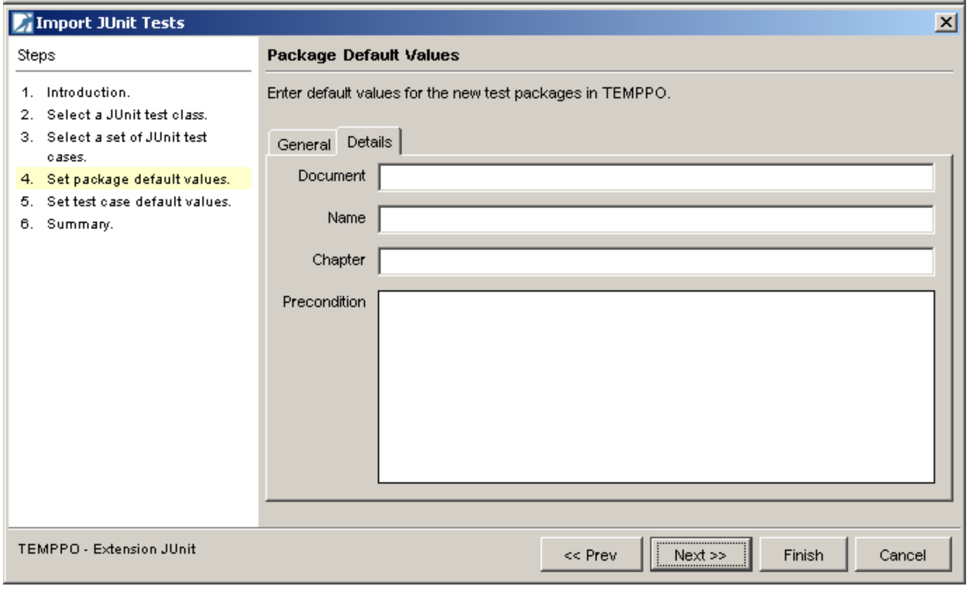
Created: <generated>

Type: Test Package

Test Category:

Description:

☑ Add to description "



Import JUnit Tests

Steps

1. Introduction.
2. Select a JUnit test class.
3. Select a set of JUnit test cases.
4. Set package default values.
5. Set test case default values.
6. Summary.

Package Default Values

Enter default values for the new test packages in TEMPPO.

General | Details

Document:

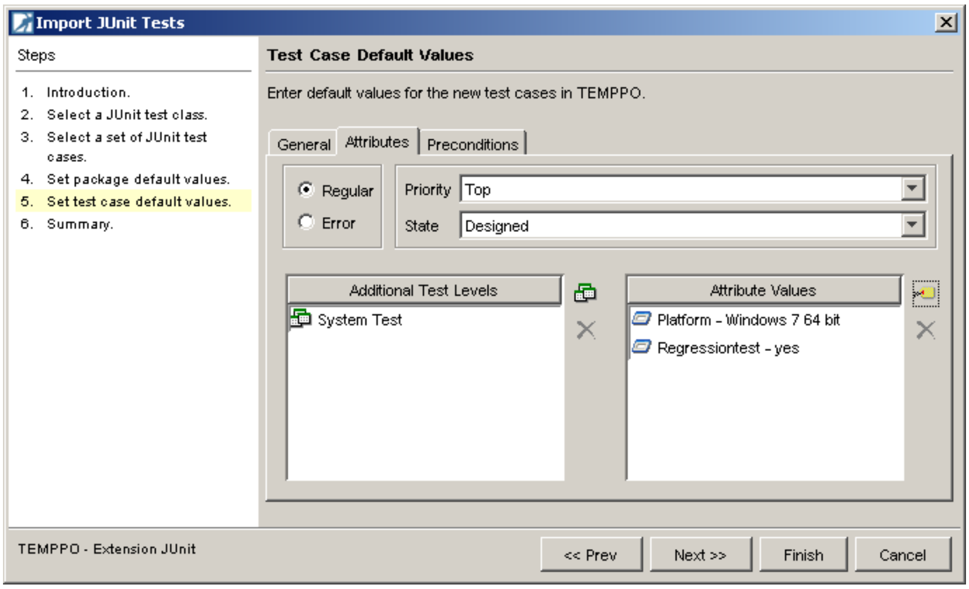
Name:

Chapter:

Precondition:

Figure 355 – Step 4: Test package default values can be entered.

In the next step, default values for JUnit test methods (i.e. TEMPPO test cases) can be entered. Note that the same tabs as for TEMPPO test cases are presented in the wizard. However, it is not possible to enter values in the "General" tab. The reason is that the only possible input field would be "Test Goal", and there is no meaningful text that could be entered there. Since the test goal must be specific for each individual test case, it does not make sense to define one common (and therefore abstract) goal for all test cases.



Import JUnit Tests

Steps

1. Introduction.
2. Select a JUnit test class.
3. Select a set of JUnit test cases.
4. Set package default values.
5. Set test case default values.
6. Summary.

Test Case Default Values

Enter default values for the new test cases in TEMPPO.

● Regular
○ Error

Priority: Top

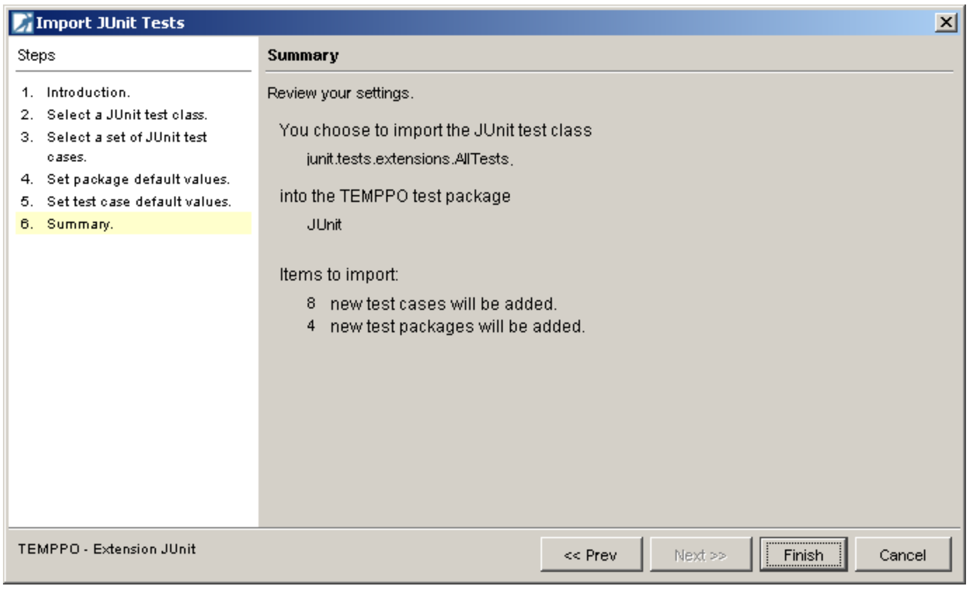
State: Designed

Additional Test Levels: System Test

Attribute Values: Platform - Windows 7 64 bit; Regressiontest - yes

TEMPPO - Extension JUnit

Figure 356 – Step 5: Test case default values can be entered.



Import JUnit Tests

Steps

1. Introduction.
2. Select a JUnit test class.
3. Select a set of JUnit test cases.
4. Set package default values.
5. Set test case default values.
6. Summary.

Summary

Review your settings.

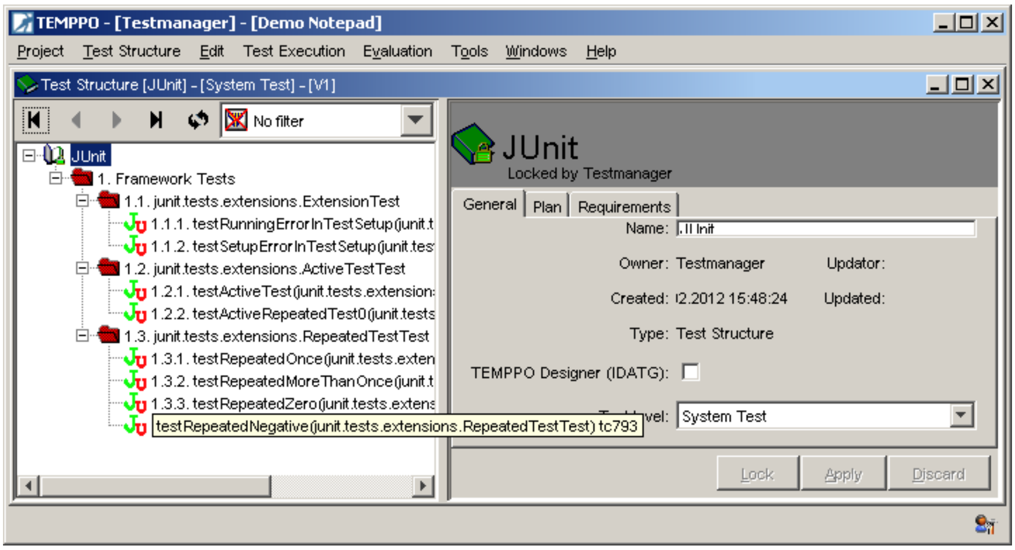
You choose to import the JUnit test class
junit.tests.extensions.AllTests.

into the TEMPPO test package
JUnit

Items to import:
8 new test cases will be added.
4 new test packages will be added.

TEMPPO - Extension JUnit

Figure 357 – Step 5: The summary of the import wizard.



JUnit
1. Framework Tests
1.1. junit.tests.extensions.ExtensionTest
1.1.1. testRunningErrorInTestSetup(junit.t
1.1.2. testSetupErrorInTestSetup(junit.tes
1.2. junit.tests.extensions.ActiveTestTest
1.2.1. testActiveTest(junit.tests.extension:
1.2.2. testActiveRepeatedTest0(junit.tests
1.3. junit.tests.extensions.RepeatedTestTest
1.3.1. testRepeatedOnce(junit.tests.exten
1.3.2. testRepeatedMoreThanOnce(junit.t
1.3.3. testRepeatedZero(junit.tests.extens
testRepeatedNegative(junit.tests.extensions.RepeatedTestTest) tc793

Name: .II Init
Owner: Testmanager Updator:
Created: I2.2012 15:48:24 Updated:
Type: Test Structure
TEMPPO Designer (IDATG): □
T... Level: System Test

Figure 358 -: Step 5: The result of the import wizard.

6.3.2 References to JUnit test cases

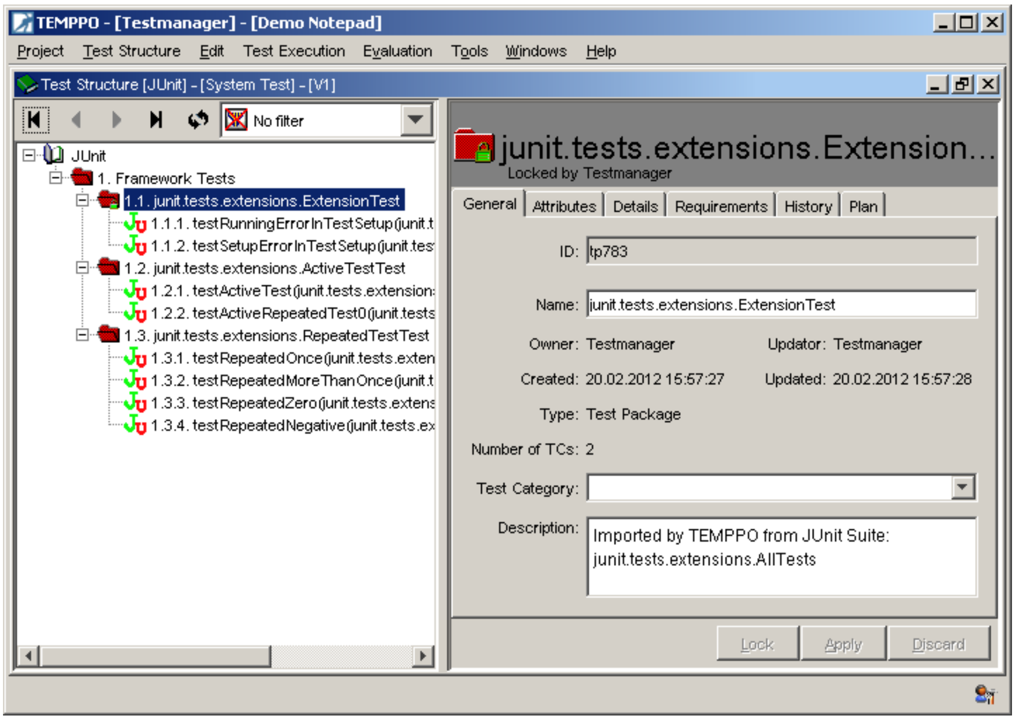
Structuring of test packages: After importing, a hierarchy of test packages and test cases according to the hierarchy of the suites defined in the JUnit test class is created. The new test packages and test cases are named according to the JUnit items, and a reference to a JUnit class and method is recorded in each test case.

Naming of test packages: If the JUnit suite object is created by defining a test class (e.g. ...new TestSuite(SimpleTest.class);), the test package is named according to this JUnit class. The naming pattern is *package.subpackage.class*.

If a JUnit suite provides a descriptive name (e.g. ...new TestSuite("All JUnit Tests");), it is used instead.

If the JUnit suite object has no name and no defined test class (e.g. ...new TestSuite();), the test package gets the current java object name of the JUnit suite (e.g. junit.framework.Suite@12345). Notice that this name doesn't refer to the JUnit suite and thus the reference is lost. Since this will disable a proper update, the JUnit suite should be always created either by giving it a name or by defining a test class.

ⓘ **It is highly recommended to create a JUnit suite by giving it a name or by defining a test class in the constructor. Otherwise the referenced Test Package in TEMPPO will get the temporary object name of the suite (e.g. junit.framework.Suite@12345). If this happens, the reference to the JUnit suite is lost and a later update won't work properly!**


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [JUnit] - [System Test] - [V1]
No filter
General Attributes Details Requirements History Plan
Lock Apply Discard

- JUnit
  - 1. Framework Tests
    - 1.1. junit.tests.extensions.ExtensionTest
      - 1.1.1. testRunningErrorInTestSetup(junit.t
      - 1.1.2. testSetupErrorInTestSetup(junit.tes
    - 1.2. junit.tests.extensions.ActiveTestTest
      - 1.2.1. testActiveTest(junit.tests.extension
      - 1.2.2. testActiveRepeatedTest0(junit.tests
    - 1.3. junit.tests.extensions.RepeatedTestTest
      - 1.3.1. testRepeatedOnce(junit.tests.exten
      - 1.3.2. testRepeatedMoreThanOnce(junit.t
      - 1.3.3. testRepeatedZero(junit.tests.extens
      - 1.3.4. testRepeatedNegative(junit.tests.ex

junit.tests.extensions.Extension...
Locked by Testmanager

ID: tp783

Name: junit.tests.extensions.ExtensionTest

Owner: Testmanager Updator: Testmanager

Created: 20.02.2012 15:57:27 Updated: 20.02.2012 15:57:28

Type: Test Package

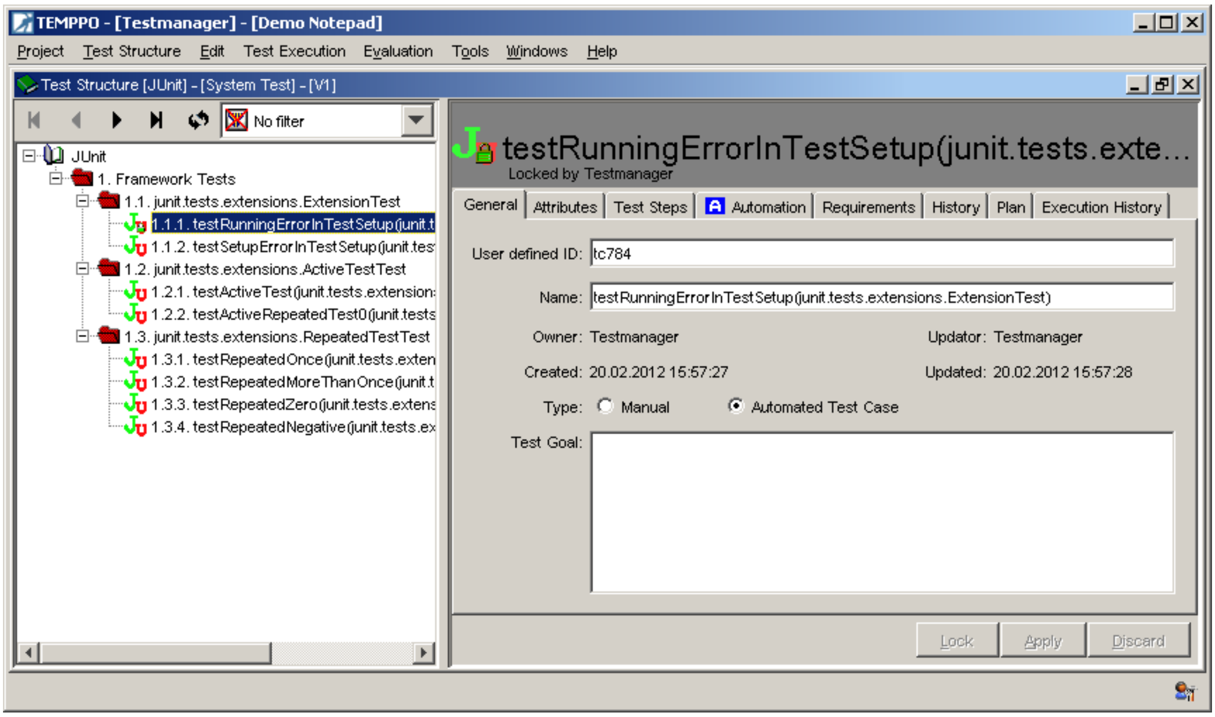
Number of TCs: 2

Test Category:

Description: Imported by TEMPPO from JUnit Suite: junit.tests.extensions.AllTests

Figure 359 – Step 5: Names of test packages (JUnit test suites).

Naming of test cases: A JUnit test case is implemented as a method of a sub-class of *TestCase* and usually has a name starting with “test”. The test cases in TEMPPO will be named according to the JUnit method name, followed by the class name in parentheses: *testMethod(SubClassOfTestCase)*.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [JUnit] - [System Test] - [V1]
No filter
General Attributes Test Steps Automation Requirements History Plan Execution History
Lock Apply Discard

- JUnit
  - 1. Framework Tests
    - 1.1. junit.tests.extensions.ExtensionTest
      - 1.1.1. testRunningErrorInTestSetup(junit.t
      - 1.1.2. testSetupErrorInTestSetup(junit.tes
    - 1.2. junit.tests.extensions.ActiveTestTest
      - 1.2.1. testActiveTest(junit.tests.extension
      - 1.2.2. testActiveRepeatedTest0(junit.tests
    - 1.3. junit.tests.extensions.RepeatedTestTest
      - 1.3.1. testRepeatedOnce(junit.tests.exten
      - 1.3.2. testRepeatedMoreThanOnce(junit.t
      - 1.3.3. testRepeatedZero(junit.tests.extens
      - 1.3.4. testRepeatedNegative(junit.tests.ex

testRunningErrorInTestSetup(junit.tests.exte...
Locked by Testmanager

User defined ID: tc784

Name: testRunningErrorInTestSetup(junit.tests.extensions.ExtensionTest)

Owner: Testmanager Updator: Testmanager

Created: 20.02.2012 15:57:27 Updated: 20.02.2012 15:57:28

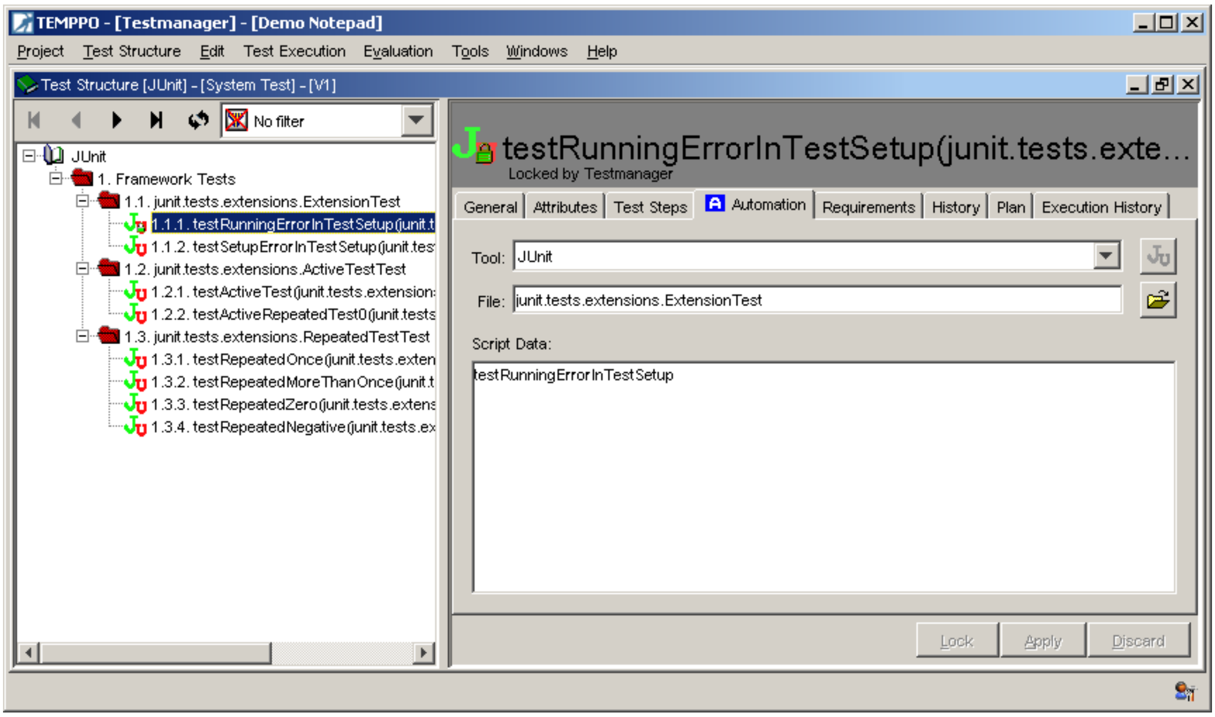
Type: ○ Manual ● Automated Test Case

Test Goal:

Figure 360 – Step 5: Names of test cases (JUnit test methods).

ⓘ **Basically you are free to edit and change the automatically created structure and the names of test packages and test cases. However, the update function (see below) relies on the names to maintain the relationship to the JUnit test classes.**

Referencing the JUnit class file: In the *Automation* panel, each test case contains a reference to the file where its code is defined. This is a JUnit class file and its path is written like in Java. E.g., a reference that reads *junit.samples.VectorTest* refers to a class file *...\junit\samples\VectorTest.class*.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [JUnit] - [System Test] - [V1]
No filter
JUnit
1. Framework Tests
1.1. junit.tests.extensions.ExtensionTest
1.1.1. testRunningErrorInTestSetup(junit.t
1.1.2. testSetupErrorInTestSetup(junit.tes
1.2. junit.tests.extensions.ActiveTestTest
1.2.1. testActiveTest(junit.tests.extension:
1.2.2. testActiveRepeatedTest0(junit.tests
1.3. junit.tests.extensions.RepeatedTestTest
1.3.1. testRepeatedOnce(junit.tests.exten
1.3.2. testRepeatedMoreThanOnce(junit.t
1.3.3. testRepeatedZero(junit.tests.extens
1.3.4. testRepeatedNegative(junit.tests.ex
testRunningErrorInTestSetup(junit.tests.exte...
Locked by Testmanager
General Attributes Test Steps Automation Requirements History Plan Execution History
Lock Apply Discard

Tool: JUnit

File: junit.tests.extensions.ExtensionTest

Script Data:

testRunningErrorInTestSetup

Figure 361 – Step 5: Reference to the JUnit class file.

Referencing the JUnit test method: In addition, each TEMPPO test case knows the name of its JUnit test method. This information is kept in the *Script Data* field of the *Automation* panel (see above).

ⓘ **The values of the JUnit class file and the test method name are editable for each test case. However, if you change these values, TEMPPO will no longer be able to find the JUnit test method. In addition, the synchronization with a newer version of the JUnit class file will be considerably hampered (see update function below).**

6.4 Update JUnit test cases

JUnit test suites and their test methods evolve over time. The update function makes it possible to synchronize these changes with the TEMPPO test structure contents. As mentioned above, the names of test packages and test cases are used as references to the corresponding JUnit suites and test methods. So in order to make the update work as you would expect it, do not change any of the following items:

- Names of test packages
- Names of test cases
- Entries in test case Automation tabs

6.4.1 Description and steps

The update function allows you to update the JUnit test cases as well as their hierarchical structure in accordance with changes in the corresponding JUnit test classes. There are two tasks for the update function:

- Add new test cases and test packages to the test structure
- Remove obsolete test cases and test packages

You can access the update function through the TEMPPO menu *Tools>JUnit>Update...* This menu is activated only when a test package is selected. There is no means to update the whole test structure at once.

A wizard guides you through the steps necessary to update a TEMPPO test package. The following steps are encompassed:

2. *Introduction*: Explains the steps of the update procedure

Test Class Selection: Provides a list of JUnit test classes you may update. Only classes found in the *classpath* are listed here (see 6.2).

Add New Test Cases: Provides the hierarchical structure of new test cases, i.e. such that were found in the suite defined by the selected JUnit class but not in the selected test package. Select all test cases that should be imported into TEMPPO.

Remove Obsolete Test Cases: Provides the hierarchical structure of obsolete test cases, i.e. such that were found in the selected test package but not in the suite defined by the selected JUnit class. Select all test cases that should be removed from the selected test package.

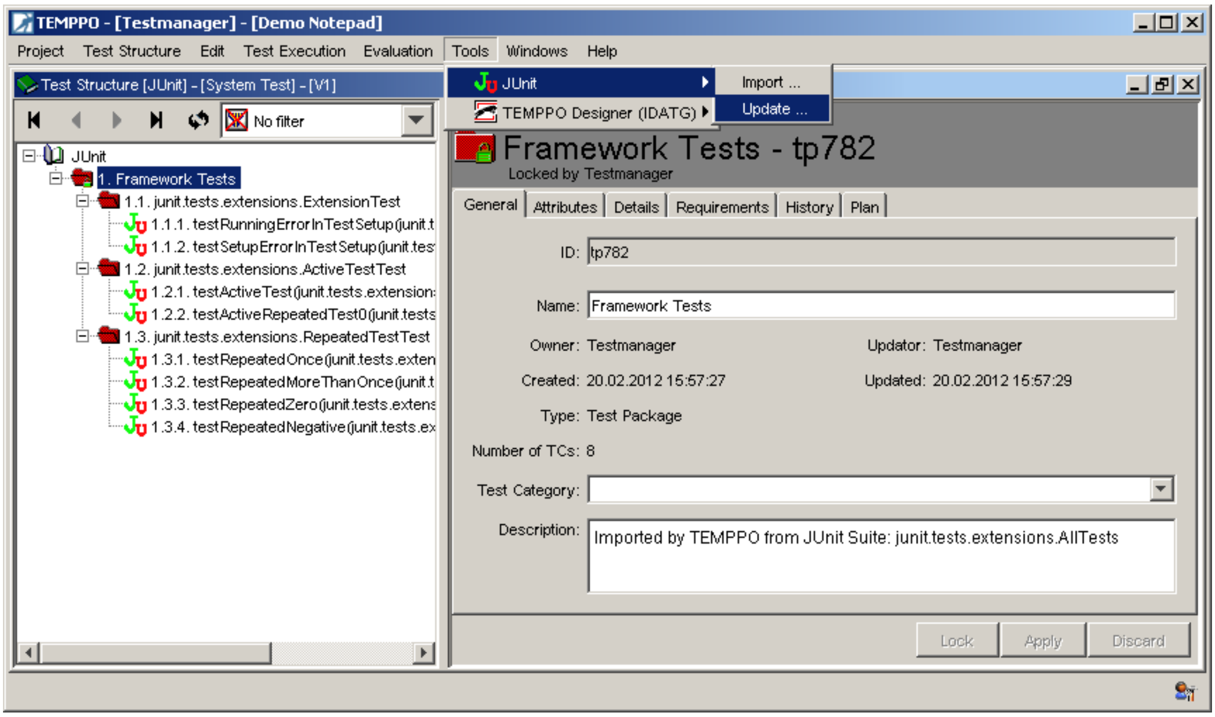
Package Default Values: Allows to set default values for all new test packages

Test Case Default Values: Allows setting default values for all new test cases.

Summary: Shows a list of your choices from previous steps.

Here is a sample update sequence. To simulate some changes, we apply the following before the update procedure is started:

- remove test package *Framework Tests* (to simulate a new test package)
- create a new test case in package *junit.samples.VectorTest* (to simulate an obsolete test case)



- JUnit
  - 1. Framework Tests
    - 1.1. junit.tests.extensions.ExtensionTest
      - 1.1.1. testRunningErrorInTestSetup(junit.t
      - 1.1.2. testSetupErrorInTestSetup(junit.tes
    - 1.2. junit.tests.extensions.ActiveTestTest
      - 1.2.1. testActiveTest(junit.tests.extension:
      - 1.2.2. testActiveRepeatedTest0(junit.tests
    - 1.3. junit.tests.extensions.RepeatedTestTest
      - 1.3.1. testRepeatedOnce(junit.tests.exten
      - 1.3.2. testRepeatedMoreThanOnce(junit.t
      - 1.3.3. testRepeatedZero(junit.tests.extens
      - 1.3.4. testRepeatedNegative(junit.tests.ex

Framework Tests - tp782
Locked by Testmanager

ID: tp782

Name: Framework Tests

Owner: Testmanager   Updator: Testmanager

Created: 20.02.2012 15:57:27   Updated: 20.02.2012 15:57:29

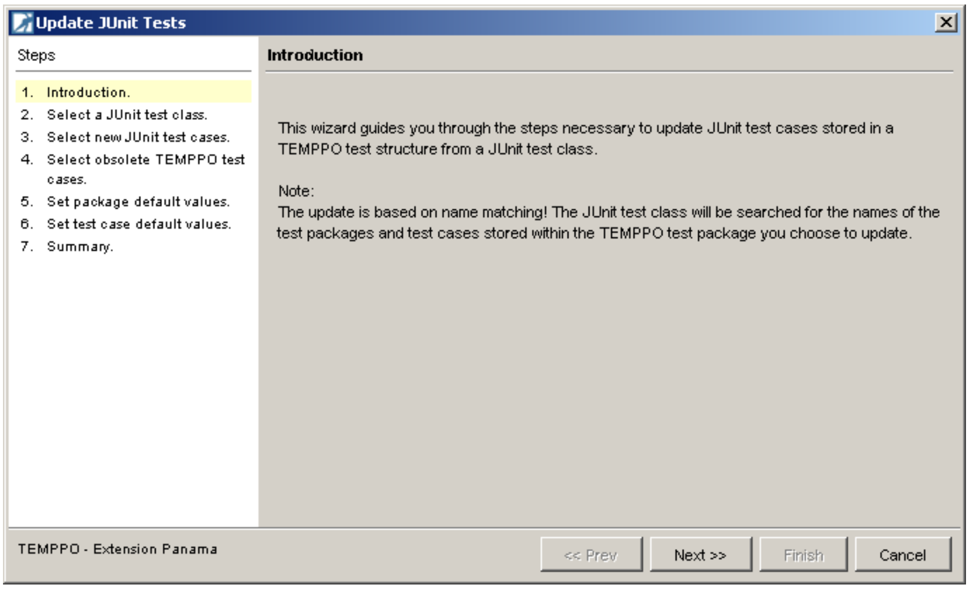
Type: Test Package

Number of TCs: 8

Test Category:

Description: Imported by TEMPPO from JUnit Suite: junit.tests.extensions.AllTests

Figure 362 – Starting the update procedure

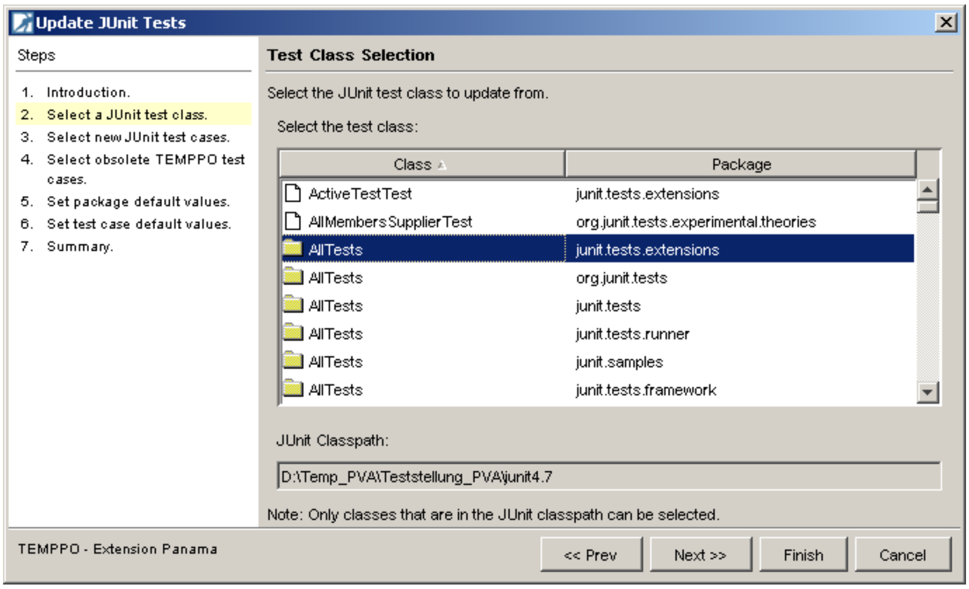


1. Introduction.
2. Select a JUnit test class.
3. Select new JUnit test cases.
4. Select obsolete TEMPPO test cases.
5. Set package default values.
6. Set test case default values.
7. Summary.

This wizard guides you through the steps necessary to update JUnit test cases stored in a TEMPPO test structure from a JUnit test class.

Note:
The update is based on name matching! The JUnit test class will be searched for the names of the test packages and test cases stored within the TEMPPO test package you choose to update.

Figure 363 – Step 1: General description of the following steps.



1. Introduction.
2. Select a JUnit test class.
3. Select new JUnit test cases.
4. Select obsolete TEMPPO test cases.
5. Set package default values.
6. Set test case default values.
7. Summary.

Test Class Selection

Select the JUnit test class to update from.

Select the test class:

Class	Package
ActiveTestTest	junit.tests.extensions
AllMembersSupplierTest	org.junit.tests.experimental.theories
AllTests	junit.tests.extensions
AllTests	org.junit.tests
AllTests	junit.tests
AllTests	junit.tests.runner
AllTests	junit.samples
AllTests	junit.tests.framework

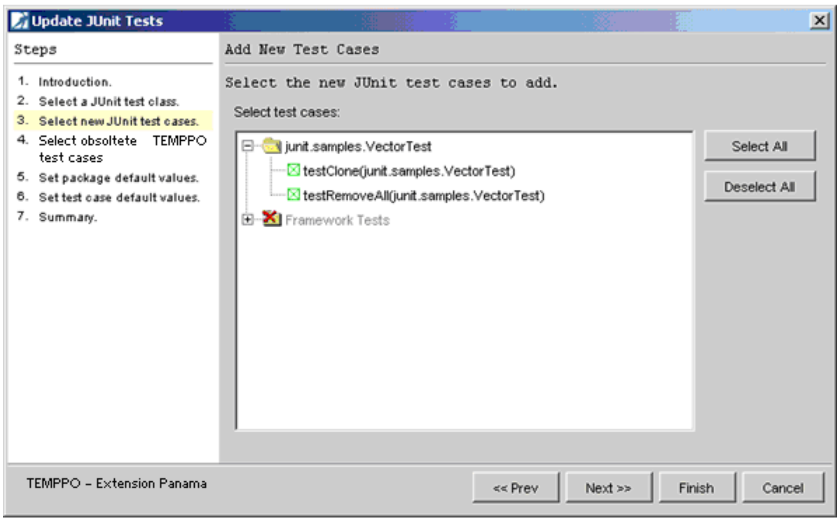
JUnit Classpath:

D:\Temp_PVA\Teststellung_PVA\junit4.7

Note: Only classes that are in the JUnit classpath can be selected.

Figure 364 – Step 2: The correct JUnit test class is automatically selected.

In the next step, new test packages and test cases will be detected. Since we removed two test cases in the import step (see Fig. 90), they will be detected as new. We also removed test package *Framework Tests* (see above), so it will also be detected as a new JUnit test suite.



1. Introduction.
2. Select a JUnit test class.
3. Select new JUnit test cases.
4. Select obsoltete TEMPPO test cases
5. Set package default values.
6. Set test case default values.
7. Summary.

Add New Test Cases

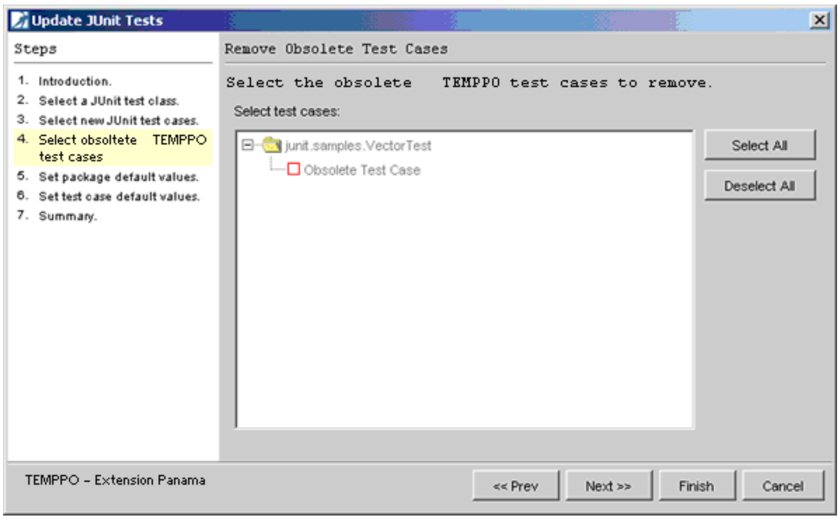
Select the new JUnit test cases to add.

Select test cases:

- junit.samples.VectorTest
  - ☒ testClone(junit.samples.VectorTest)
  - ☒ testRemoveAll(junit.samples.VectorTest)
- Framework Tests

Figure 365 – Step 3: Two new test cases and a new test package are detected automatically.

Now, the test case *ObsoleteTestCase* that we created just before (see above) will be detected as obsolete, because it is not contained in the JUnit test class *junit.samples.VectorTest*.



Update JUnit Tests

Steps
1. Introduction.
2. Select a JUnit test class.
3. Select new JUnit test cases.
4. Select obsoltete TEMPPO test cases
5. Set package default values.
6. Set test case default values.
7. Summary.

Remove Obsolete Test Cases

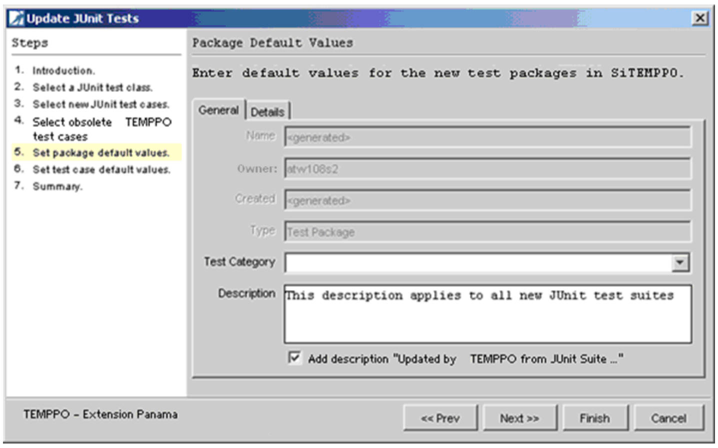
Select the obsolete TEMPPO test cases to remove.

Select test cases:

- junit.samples.VectorTest
  - □ Obsolete Test Case

Figure 366 – Step 4: An obsolete test case was detected and we mark it for being removed.

In the next steps, default values for JUnit test suites and test methods can be entered, just like in the import procedure.



Update JUnit Tests

Steps
1. Introduction.
2. Select a JUnit test class.
3. Select new JUnit test cases.
4. Select obsolete TEMPPO test cases
5. Set package default values.
6. Set test case default values.
7. Summary.

Package Default Values

Enter default values for the new test packages in SiTEMPPO.

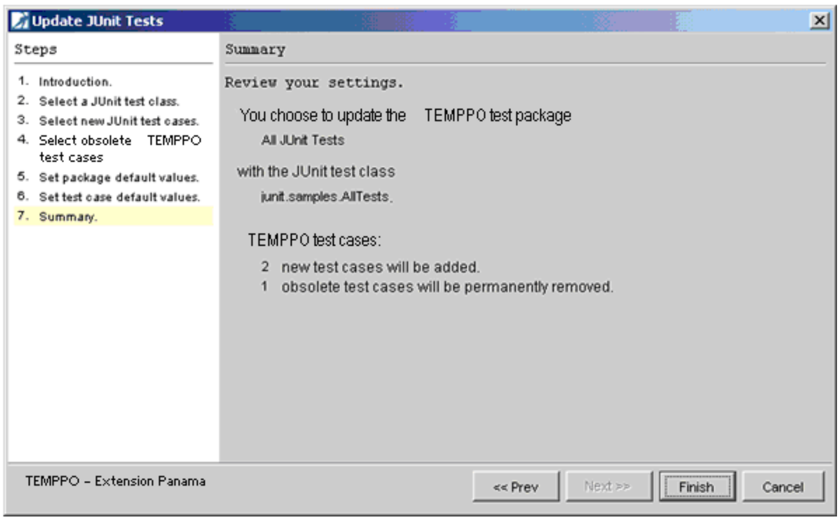
General | Details

Name: <generated>
Owner: atw108s2
Created: <generated>
Type: Test Package
Test Category:
Description: This description applies to all new JUnit test suites

☑ Add description "Updated by TEMPPO from JUnit Suite ..."

Figure 367 – Step 5: Test package default values can be entered.

In step 6 Test case values can be entered.



Update JUnit Tests

Steps
1. Introduction.
2. Select a JUnit test class.
3. Select new JUnit test cases.
4. Select obsolete TEMPPO test cases
5. Set package default values.
6. Set test case default values.
7. Summary.

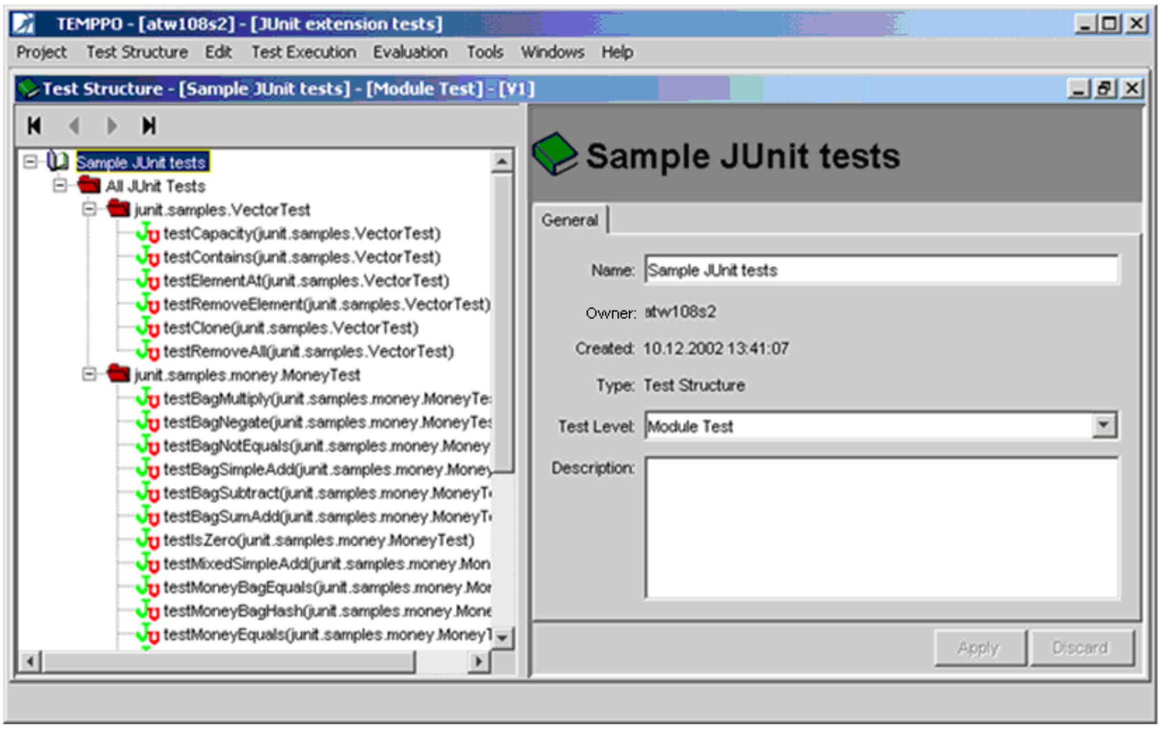
Summary

Review your settings.

You choose to update the TEMPPO test package
All JUnit Tests
with the JUnit test class
junit.samples.AllTests.

TEMPPO test cases:
2 new test cases will be added.
1 obsolete test cases will be permanently removed.

Figure 368 – Step 5: The summary of the update wizard.



TEMPPO - [atw108s2] - [JUnit extension tests]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Structure - [Sample JUnit tests] - [Module Test] - [V1]
Sample JUnit tests
General
Apply
Discard

- Sample JUnit tests
  - All JUnit Tests
    - junit.samples.VectorTest
      - testCapacity(junit.samples.VectorTest)
      - testContains(junit.samples.VectorTest)
      - testElementAt(junit.samples.VectorTest)
      - testRemoveElement(junit.samples.VectorTest)
      - testClone(junit.samples.VectorTest)
      - testRemoveAll(junit.samples.VectorTest)
    - junit.samples.money.MoneyTest
      - testBagMultiply(junit.samples.money.MoneyTe:
      - testBagNegate(junit.samples.money.MoneyTes
      - testBagNotEquals(junit.samples.money.Money
      - testBagSimpleAdd(junit.samples.money.Money
      - testBagSubtract(junit.samples.money.MoneyT
      - testBagSumAdd(junit.samples.money.MoneyT
      - testIsZero(junit.samples.money.MoneyTest)
      - testMixedSimpleAdd(junit.samples.money.Mon
      - testMoneyBagEquals(junit.samples.money.Mor
      - testMoneyBagHash(junit.samples.money.Mone
      - testMoneyEquals(junit.samples.money.Money1

Name: Sample JUnit tests
Owner: atw108s2
Created: 10.12.2002 13:41:07
Type: Test Structure
Test Level: Module Test
Description:

Figure 369 – Step 5: The result of the update wizard.

6.4.2 References to JUnit test cases

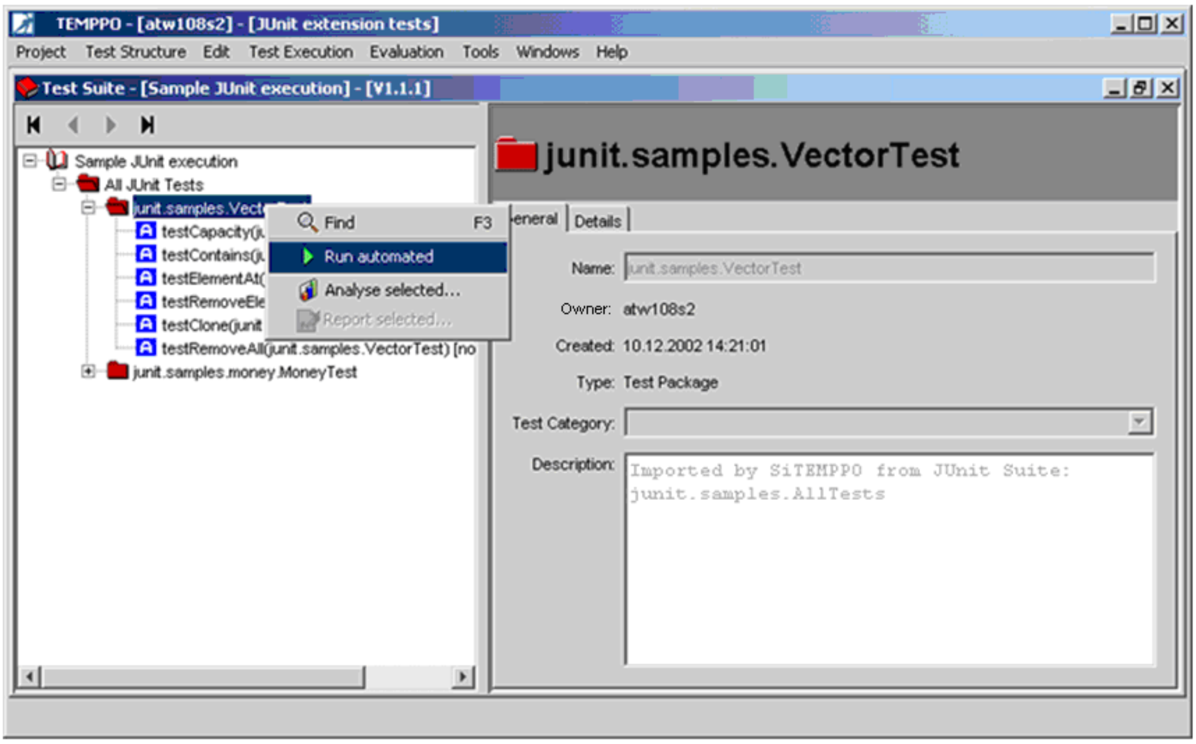
The structure of test packages and test cases as well as their names are the same as with the import procedure. Updating behaves in conformance with importing and relies on the naming patterns described above.

6.5 Test Execution

6.5.1 Description and Steps

Execution allows you to run the JUnit test cases from TEMPPO and to collect the test execution results in the TEMPPO database for further analysis.

Like any other automated test cases, you can execute JUnit test cases by creating a test suite and running the selected test cases or whole test packages (from context menu or menu: *Test Execution/Run automated*).



Name: junit.samples.VectorTest

Owner: atw108s2

Created: 10.12.2002 14:21:01

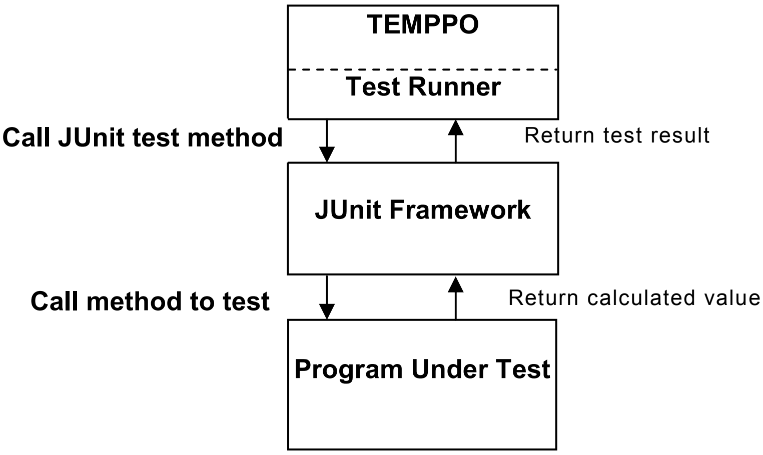
Type: Test Package

Test Category:

Description: Imported by SiTEMPPO from JUnit Suite: junit.samples.AllTests

Figure 370 – Executing JUnit test cases.

Running a JUnit test case involves three levels of execution: (1) **TEMPPO**, where the JUnit test case class is instantiated and the corresponding method is called, (2) **JUnit**, where the actual result is compared to the expected result in the test method, and (3) the classes of the **program under test** that is executed during the test. To run the tests, TEMPPO provides its own *TestRunner* that calls the JUnit test methods and collects their results. Thus, for executing JUnit tests, TEMPPO can be used interchangeably with other JUnit test runners, like `junit.swingui.TestRunner`.



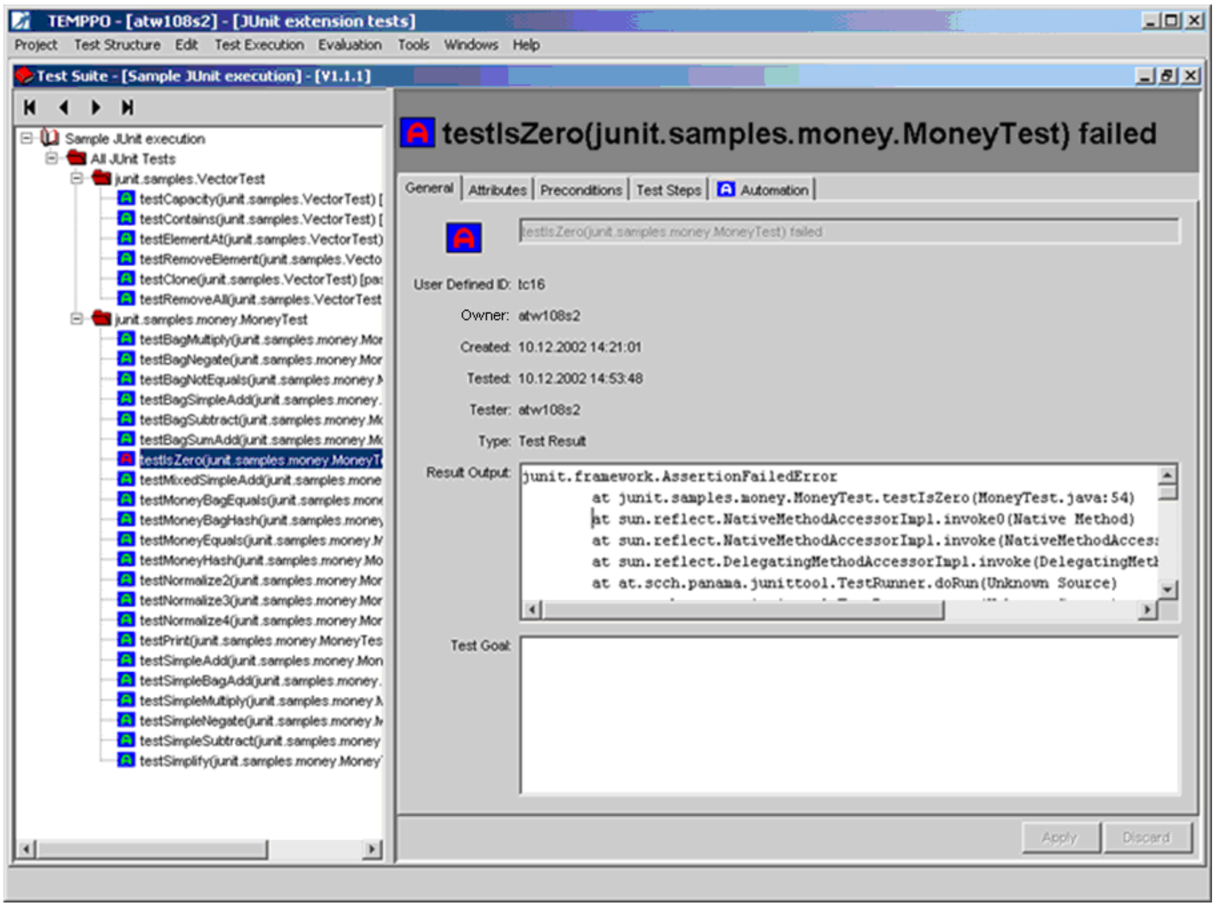
ⓘ **Note: To execute the JUnit tests, the classpath has to contain not only contain the JUnit test classes but also the classes of the program under test.**

6.5.2 Interpreting the Test Results

The execution of JUnit test cases yields one of the three results listed in Table 4. In case of a failure or blockage, a description of the reason and the stack trace showing the position where the failure occurred is given in the *Result Output* field in the panel *General* of the test case as shown in Fig. 108.

Icon	**TEMPPO Result**	**JUnit Result**	**Description**
A	PASSED	-	The test case was executed and the actual result returned by the program under test matched the expected result as specified in the JUnit test case.
A	FAILED	Failure	The test case was executed and the actual result returned by the program under test did *not* match the expected result as specified in the JUnit test case.
A	BLOCKED	Error	The execution of the test case had errors and no results could be compared. There are basically two possibilities: The call from TEMPPO to the JUnit test class failed. E.g. because the JUnit class did not exist in the classpath, the specified test method did not exist, or the class/method were not implemented according to the JUnit interface The execution of the JUnit test case had errors. E.g. the JUnit test case or the program under test threw an (uncaught) exception or even the whole program under test crashed.

Table 4 – Possible test execution results


TEMPPO - [atw108s2] - [JUnit extension tests]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Suite - [Sample JUnit execution] - [V1.1.1]
General
Attributes
Preconditions
Test Steps
Automation
Apply
Discard

Sample JUnit execution
All JUnit Tests
junit.samples.VectorTest
testCapacity(junit.samples.VectorTest) [
testContains(junit.samples.VectorTest) [
testElementAt(junit.samples.VectorTest)
testRemoveElement(junit.samples.Vecto
testClone(junit.samples.VectorTest) [pa
testRemoveAll(junit.samples.VectorTest
junit.samples.money.MoneyTest
testBagMultiply(junit.samples.money.Mo
testBagNegate(junit.samples.money.Mo
testBagNotEquals(junit.samples.money.M
testBagSimpleAdd(junit.samples.money.
testBagSubtract(junit.samples.money.Mo
testBagSumAdd(junit.samples.money.Mo
testIsZero(junit.samples.money.MoneyT
testMixedSimpleAdd(junit.samples.mone
testMoneyBagEquals(junit.samples.mon
testMoneyBagHash(junit.samples.money
testMoneyEquals(junit.samples.money.M
testMoneyHash(junit.samples.money.Mo
testNormalize2(junit.samples.money.Mo
testNormalize3(junit.samples.money.Mo
testNormalize4(junit.samples.money.Mo
testPrint(junit.samples.money.MoneyTes
testSimpleAdd(junit.samples.money.Mon
testSimpleBagAdd(junit.samples.money.
testSimpleMultiply(junit.samples.money.M
testSimpleNegate(junit.samples.money.M
testSimpleSubtract(junit.samples.money
testSimplify(junit.samples.money.Money

testIsZero(junit.samples.money.MoneyTest) failed

testIsZero(junit.samples.money.MoneyTest) failed

User Defined ID: tc16
Owner: atw108s2
Created: 10.12.2002 14:21:01
Tested: 10.12.2002 14:53:48
Tester: atw108s2
Type: Test Result
Result Output:

```
junit.framework.AssertionFailedError
        at junit.samples.money.MoneyTest.testIsZero(MoneyTest.java:54)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAcces
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMet
        at at.scch.panama.junittool.TestRunner.doRun(Unknown Source)
```

Test Goal:

Figure 371 – Failed test case: result message and stack trace

6.6 Known Issues

Unsupported JUnit extensions

Currently, only standard JUnit test classes are supported, i.e. JUnit test cases are expected to be an instance of `junit.framework.TestCase` and JUnit suites are expected to be an instance of `junit.framework.TestSuite`.

Performance of removing obsolete test cases

When a larger number (more than 10) of obsolete test cases was selected for removing in the *Update* wizard, you may experience a delay of more than 30 seconds after finishing the wizard dialog.

Malicious JUnit test cases

Executing a malicious test case may have a negative effect on the TEMPPO application, which runs in the same VirtualMachine as the JUnit test execution. Currently no means are provided for setting an execution timeout or to manually cancel the test execution.

Reloading JUnit classes

Once the unit test classes were loaded by the Java VM (e.g. when they are executed), they will not be reloaded until TEMPPO is restarted.

Classpath for JUnit test cases

The classpath containing the JUnit test cases can only be set once for each TEMPPO session and not separately for each test structure or TEMPPO project.

If the classpath contains two class files with the same fully qualified name the file with the newest build date will be loaded.

Anonymous inner classes

By statically creating the tests using anonymous inner classes that are not public, the tests from this suite will not be executeable by TEMPPO. With a JUnit TestRunner, this construction works because there JUnit suites are executed as a whole whereas TEMPPO tries to instantiate and execute each test case by itself.
Sample code:

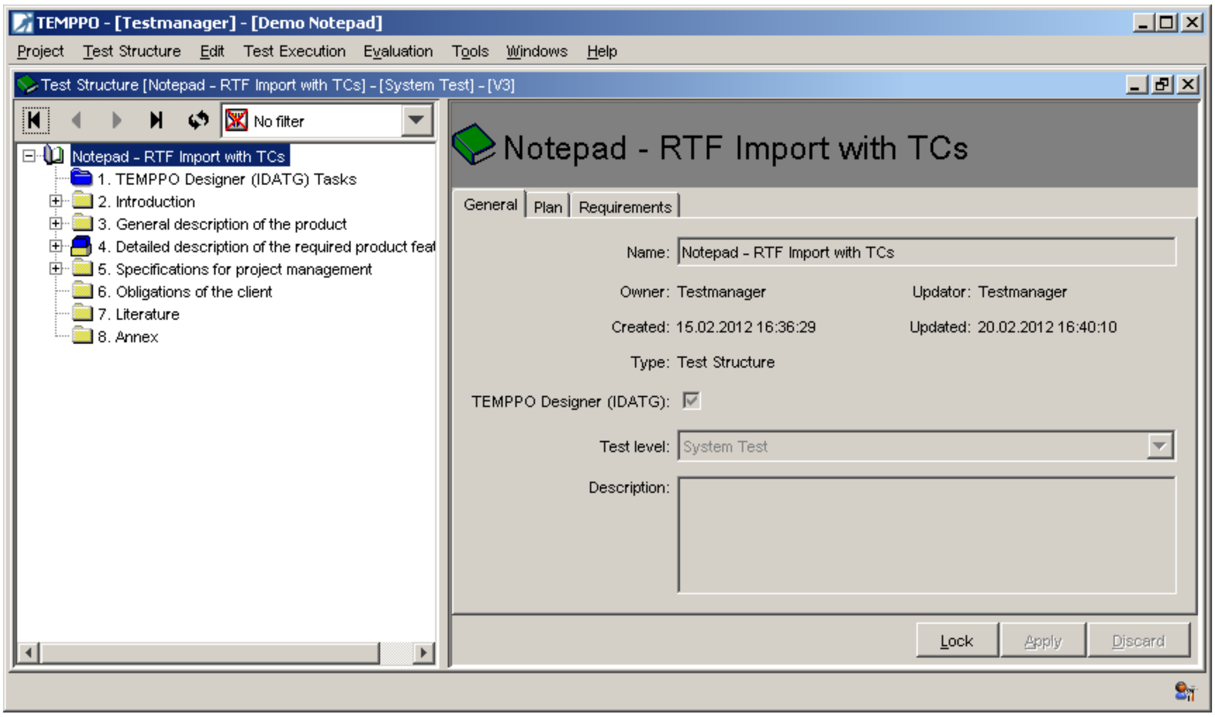
```
public class MyTestPkgWithStatSuite extends TestCase {
 public MyTestPkgWithStatSuite(String name) {super(name);}
 public static Test suite() {
   TestSuite suite= new TestSuite("MyTestPkgWithStatSuite");
   suite.addTest( new MyTestPkgWithStatSuite("testSomethingRight") {
     protected void runTest() { testSomethingRight(); }
   } );
   return suite;
   }
 public void testSomethingRight() {assertTrue(true)}
}
```

7 TEMPPO Designer (IDATG)

TEMPPO Designer (IDATG) is your gateway to professional and cost-efficient test automation. While up to now test teams had to spend a huge effort for the implementation and maintenance of automated test cases, TEMPPO Designer (IDATG) aims at generating test cases based on a simple model of the application. The advantage is obvious: Instead of having to write test scripts from scratch and manually adapt each of them whenever there is a change in the tested application, all that is needed now is keeping the model up-to-date. The costs for test maintenance are reduced dramatically thus leading to shorter test cycles and better product quality.

The unique TEMPPO Designer (IDATG) tool combines user-friendly test specification editors with a patented test case generator that employs a wide range of systematic test methods. TEMPPO Designer (IDATG) supports a number of popular GUI testing tools but can also generate scripts for other test interfaces.

For using TEMPPO Designer (IDATG) from TEMPPO, it is necessary to tick the checkbox **TEMPPO Designer (IDATG)** on tab **General** of the test structure's root.


TEMPPO - [Testmanager] - [Demo Notepad]
Project
Test Structure
Edit
Test Execution
Evaluation
Tools
Windows
Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. TEMPPO Designer (IDATG) Tasks
2. Introduction
3. General description of the product
4. Detailed description of the required product feat
5. Specifications for project management
6. Obligations of the client
7. Literature
8. Annex
Notepad - RTF Import with TCs
General
Plan
Requirements
Lock
Apply
Discard

Name: Notepad - RTF Import with TCs

Owner: Testmanager
Updator: Testmanager

Created: 15.02.2012 16:36:29
Updated: 20.02.2012 16:40:10

Type: Test Structure

TEMPPO Designer (IDATG): ☑

Test level: System Test

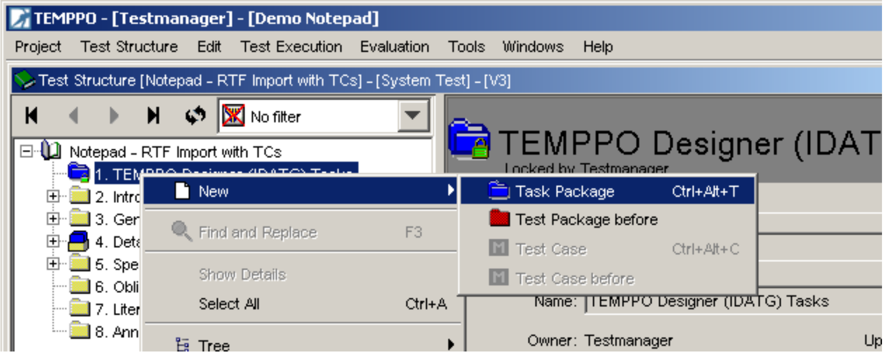
Description:

Figure 372 – TEMPPO Designer (IDATG)

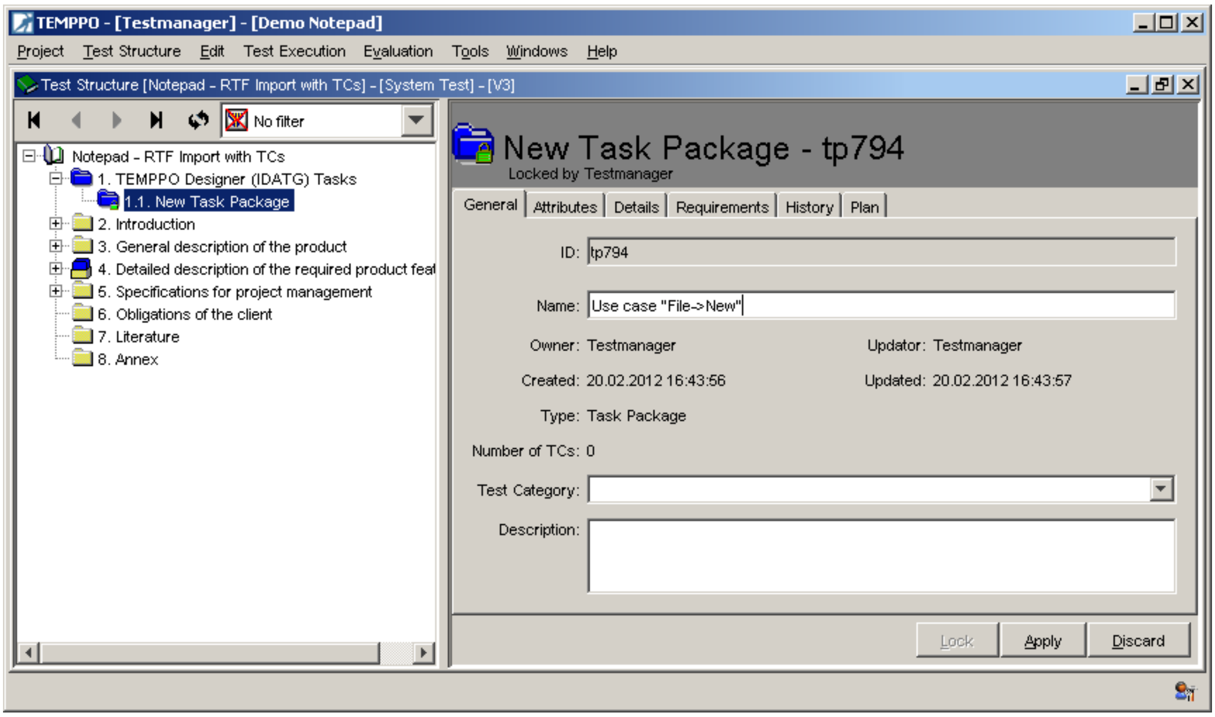
As a preparation for designing test cases in TEMPPO Designer (IDATG) it is necessary to create so-called **Task Packages** in TEMPPO. A task package is a special type of test package that has got an ID and the same attributes as a test case. You can create any kind of task package hierarchy.

7.1 Creating Task Packages

Task packages only can be created below the folder TEMPPO Designer (IDATG) Tasks by activating the menu item **New->Task Package** or **New->Task Package before**.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. TEMPPO Designer (IDATG) Tasks
2. Intro
3. Ger
4. Deta
5. Spe
6. Obli
7. Liter
8. Ann
New
Find and Replace F3
Show Details
Select All Ctrl+A
Tree
Task Package Ctrl+Alt+T
Test Package before
Test Case Ctrl+Alt+C
Test Case before
TEMPPO Designer (IDAT
Locked by Testmanager
Name: TEMPPO Designer (IDATG) Tasks
Owner: Testmanager
Up

A task package contains all properties of a test case: ID (fixed), name, owner, created, updator, updated, test category (optional), description.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. TEMPPO Designer (IDATG) Tasks
1.1. New Task Package
2. Introduction
3. General description of the product
4. Detailed description of the required product feat
5. Specifications for project management
6. Obligations of the client
7. Literature
8. Annex
New Task Package - tp794
Locked by Testmanager
General
Attributes
Details
Requirements
History
Plan
Lock
Apply
Discard

ID: tp794

Name: Use case "File->New"

Owner: Testmanager

Updator: Testmanager

Created: 20.02.2012 16:43:56

Updated: 20.02.2012 16:43:57

Type: Task Package

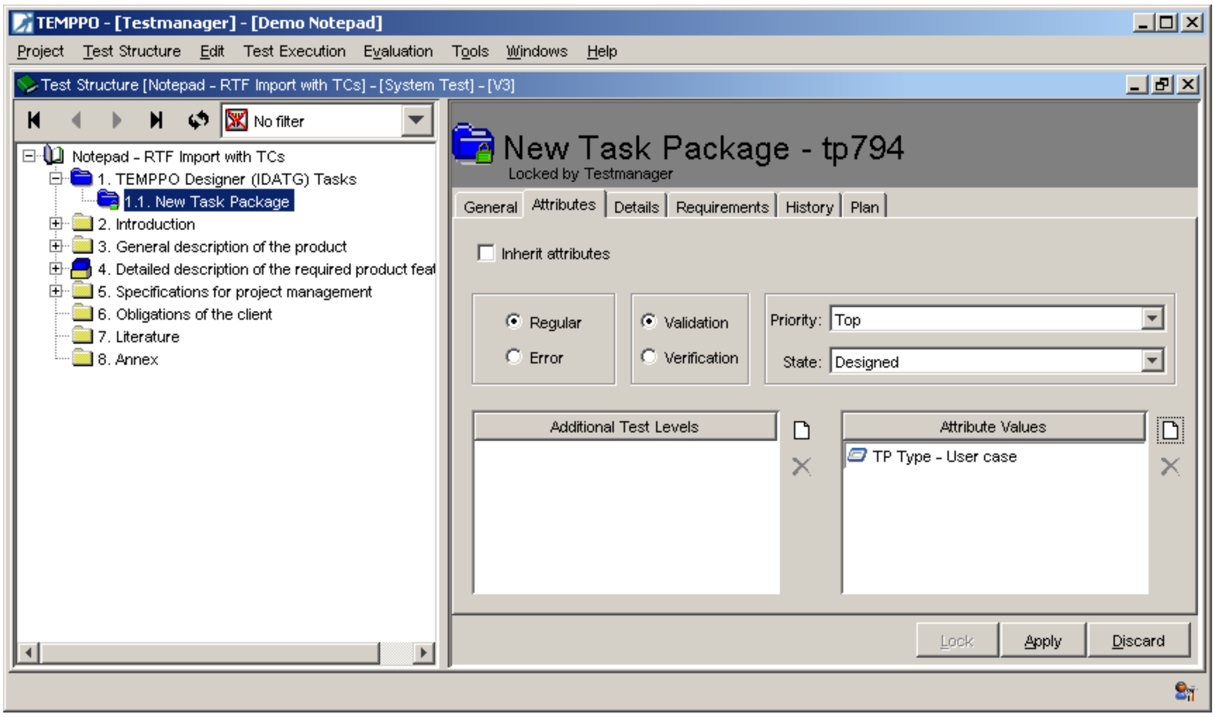
Number of TCs: 0

Test Category:

Description:

Figure 373 – Task Package – Tab general

In addition, attributes can be set which are later inherited and used for test cases generated by TEMPPO Designer (IDATG). It is necessary to assign attributes to task packages and not to test cases because test cases may be generated more than once. Each time test cases are generated again, all the previous information will be lost. Therefore the meta information of test cases generated by TEMPPO Designer (IDATG) is stored in task packages.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System Test] - [V3]
No filter
Notepad - RTF Import with TCs
1. TEMPPO Designer (IDATG) Tasks
1.1. New Task Package
2. Introduction
3. General description of the product
4. Detailed description of the required product feat
5. Specifications for project management
6. Obligations of the client
7. Literature
8. Annex
New Task Package - tp794
Locked by Testmanager
General Attributes Details Requirements History Plan
Additional Test Levels
Attribute Values
TP Type - User case
Lock Apply Discard

☐ Inherit attributes

● Regular
○ Error

● Validation
○ Verification

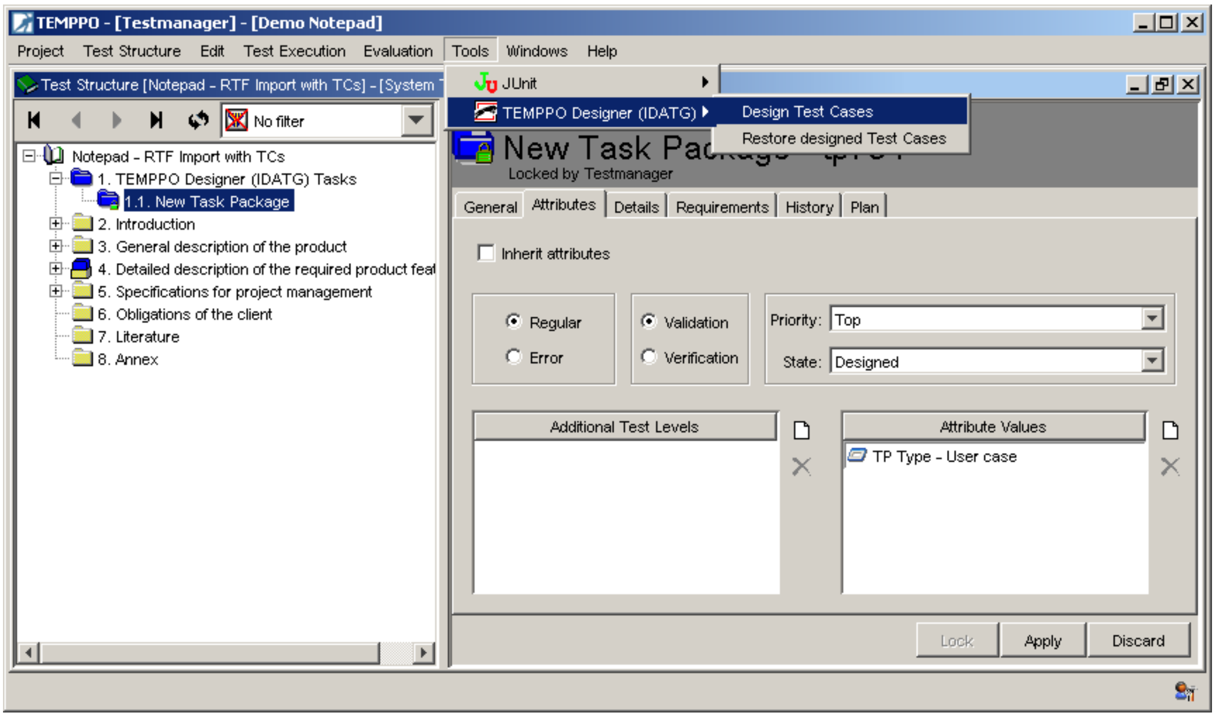
Priority: Top

State: Designed

Figure 374 – Task Package – Tab attributes

7.2 Design Test Cases

If you have finished creating some task packages, you can design and generate test cases by selecting the node "TEMPPO Designer (IDATG) Tasks" and call the menu item **Tools->TEMPPO Designer (IDATG)->Design Test Cases**.


TEMPPO - [Testmanager] - [Demo Notepad]
Project Test Structure Edit Test Execution Evaluation Tools Windows Help
Test Structure [Notepad - RTF Import with TCs] - [System
JUnit
TEMPPO Designer (IDATG)
Design Test Cases
Restore designed Test Cases
No filter
Notepad - RTF Import with TCs
1. TEMPPO Designer (IDATG) Tasks
1.1. New Task Package
2. Introduction
3. General description of the product
4. Detailed description of the required product feat
5. Specifications for project management
6. Obligations of the client
7. Literature
8. Annex
New Task Pac
Locked by Testmanager
General Attributes Details Requirements History Plan
Additional Test Levels
Attribute Values
TP Type - User case
Lock Apply Discard

☐ Inherit attributes

● Regular
○ Error

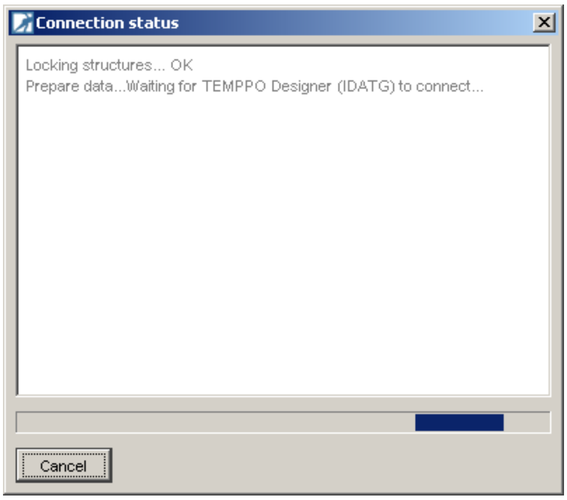
● Validation
○ Verification

Priority: Top

State: Designed

Figure 375 - Design Test cases

After activating the menu item **Design Test Cases** the window **Connection Status** (Figure 376) and TEMPPO Designer (IDATG) itself are displayed.



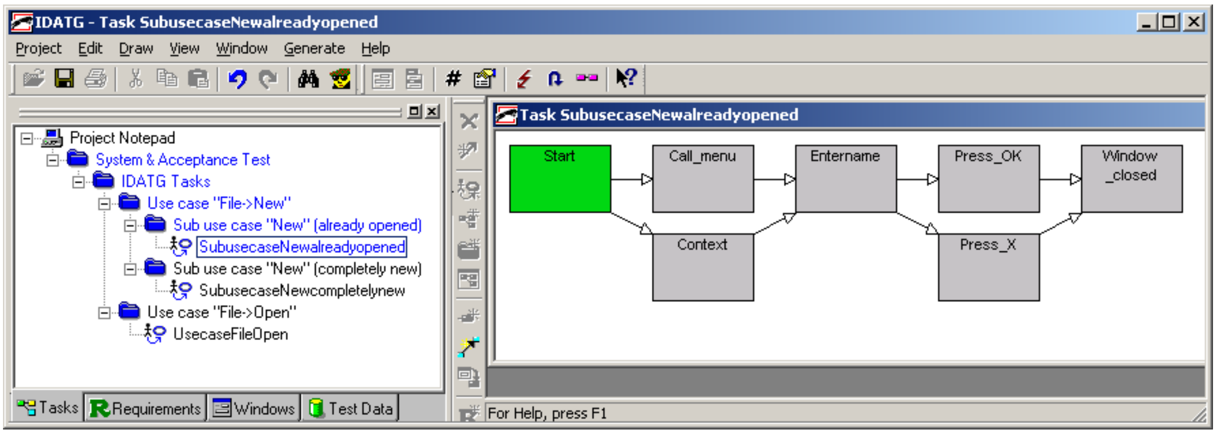
Locking structures... OK
Prepare data...Waiting for TEMPPO Designer (IDATG) to connect...

Figure 376 – Connection Status

TEMPPO Designer (IDATG) offers a huge number of features for test design, test generation, and applying several test design methods. For detailed information see the TEMPPO Designer (IDATG) User's Guide.

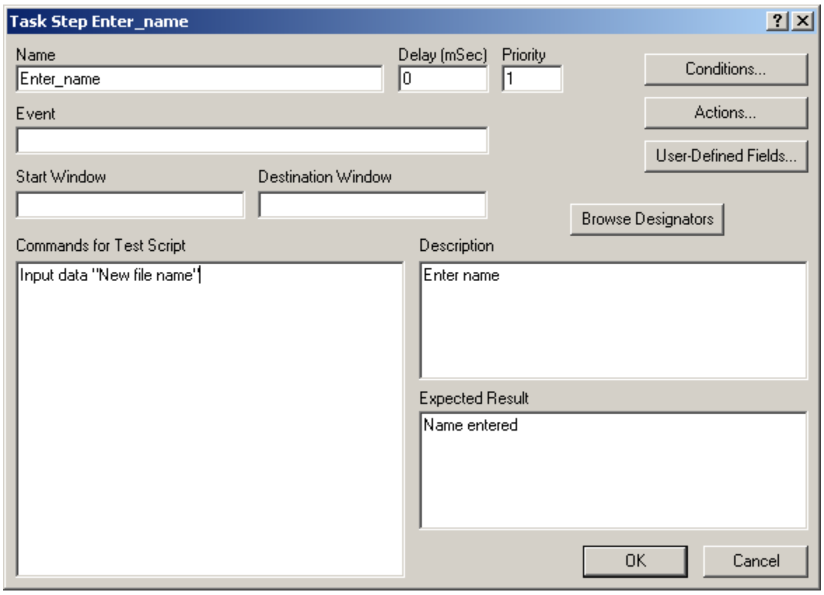
The following screen shots illustrate a simple example of designing a task flow with several steps. When TEMPPO Designer (IDATG) is opened, it displays the task packages defined in TEMPPO and automatically creates a use case for each leaf in the task package hierarchy The user has now the possibility of creating task flows consisting of several steps.

The following figure shows a modeled task flow consisting of some steps. A (task) step consists of several attributes like name, description, expected result, commands for test script, conditions etc. All this information will be used in the generated test cases. Figure 378 displays a sample step.



IDATG - Task SubusecaseNewalreadyopened
Project Edit Draw View Window Generate Help
Project Notepad
System & Acceptance Test
IDATG Tasks
Use case "File->New"
Sub use case "New" (already opened)
SubusecaseNewalreadyopened
Sub use case "New" (completely new)
SubusecaseNewcompletelynew
Use case "File->Open"
UsecaseFileOpen
Tasks
Requirements
Windows
Test Data
Task SubusecaseNewalreadyopened
Start
Call_menu
Entername
Press_OK
Window _closed
Context
Press_X
For Help, press F1

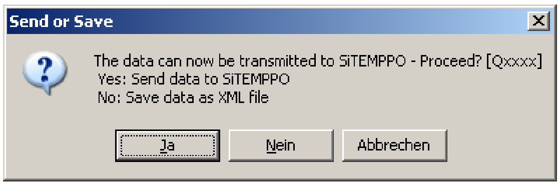
Figure 377 – Design Test Cases



Name: Enter_name
Delay (mSec): 0
Priority: 1
Event:
Start Window:
Destination Window:
Commands for Test Script: Input data "New file name"|
Description: Enter name
Expected Result: Name entered

Figure 378 – Task Flow Step

If the user has finished the modeling process, he can start TEMPPO Designer (IDATG)'s test case generator. Depending on the selected options, TEMPPO Designer (IDATG) will automatically generate test cases until the desired coverage is reached. For instance, 2 test cases would be generated for the task flow above if "Step Coverage" is selected. To import these test cases into TEMPPO, the user just has to press the "Save" button.



The data can now be transmitted to SiTEMPPO - Proceed? [Qxxxx]
Yes: Send data to SiTEMPPO
No: Save data as XML file

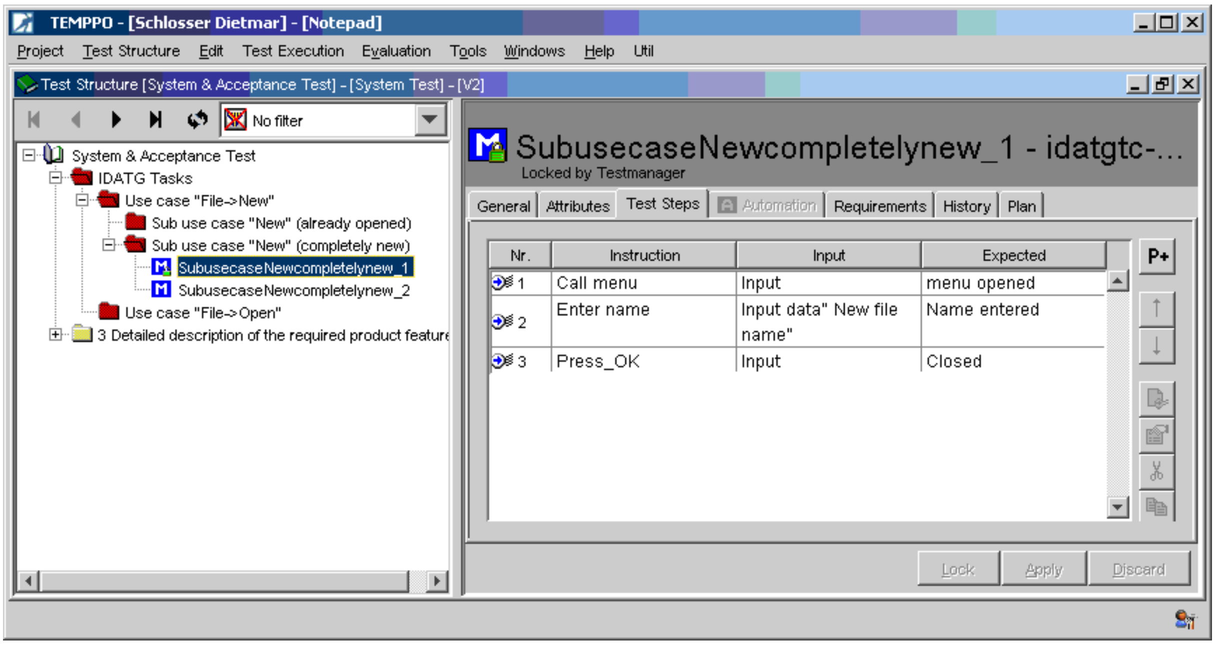
Figure 379 – Save dialog

It can be configured in settings for TEMPPO Designer, if test structure is checked-in with data of TEMPPO Designer.
The file used by TEMPPO Designer (IDATG) is saved in TEMPPO for each test structure version.

For **manual** test cases, the generated steps are displayed in TEMPPO. TEMPPO fields are mapped to TEMPPO Designer (IDATG) fields as follows:

• Instruction	=	Description
• Input	=	Commands for Test Script
• Expected	=	Expected Result

For **automated** test cases, only the path to the generated test script is displayed. A test case is treated as automated, if it has at least once been exported by TEMPPO Designer (IDATG) as a test script for an automation tool (e.g. TestPartner).



- System & Acceptance Test
  - IDATG Tasks
    - Use case "File->New"
      - Sub use case "New" (already opened)
      - Sub use case "New" (completely new)
        - SubusecaseNewcompletelynew_1
        - SubusecaseNewcompletelynew_2
    - Use case "File->Open"
  - 3 Detailed description of the required product feature

SubusecaseNewcompletelynew_1 - idatgtc-...
Locked by Testmanager

Nr.	Instruction	Input	Expected
1	Call menu	Input	menu opened
2	Enter name	Input data" New file name"	Name entered
3	Press_OK	Input	Closed

tbd

Figure 380 – Test cases generated by TEMPPO Designer (IDATG)

7.3 Updating Test Cases

Chapter 7.2 describes the creation of task packages and the design and generation of test cases with TEMPPO Designer (IDATG).

Of course, it is possible to change the existing task packages or to add new ones by calling TEMPPO Designer (IDATG) again.

7.4 Restore Designed Test Cases

If the connection between TEMPPO and TEMPPO Designer (IDATG) is lost (e.g. because the user aborts it by pressing the button **Cancel** in the window **Connection Status**), it is possible to avoid data loss by storing the current TEMPPO Designer (IDATG) project as XML file.

To restore this data and transfer it to TEMPPO, the menu item **Restore Designed Test Cases** can be used. This will cause TEMPPO Designer (IDATG) to open, to load the XML file and to synchronize it with the task package structure received from TEMPPO. By pressing “Save” the data can be transferred back to TEMPPO.

8 Command Line

TEMPPO offers following command line parameters:

Remark: No blanks are allowed between commas!

-open <project,test structure,[version],[test suite]>	, separated list of values, where the names for project, test structure must be specified and TEMPPO is opened with the latest version of the test structure. Optionally a special version and a test suite can be entered in order to open immediately a test structure version or a test suite.
-run <project,test structure,version,test suite> [-exit]	, separated list of values, where the names for project, test structure, version and test suite must be specified. All automated test cases in the specified test suite will be executed. Optionally TEMPPO is closed with the –exit parameter.
-export_teststructure <project,test structure,version,export file>	, separated list of values, where the names for project, test structure, version and export file must be specified. The whole test structure is exported to the denoted file (XML format).
-export_testsuite <project,test structure,version,test suite,export file>	, separated list of values, where the names for project, test structure, version, test suite and export file must be specified. The whole test suite is exported to the denoted file (XML format).
-check_locks	Checks, if there are locks by current user. Considered in conjunction with option "export_teststructure" and "export_testsuite"
-copy_testsuite <project,test structure,version,existing test suite,new test suite name,flag actual result,flag bugid>	, separated list of values, where the names for project, test structure, version, existing test suite, new test suite name and flag 'actual result' and 'bug id' must be specified with 'true' or 'false'. A new test suite is created with settings of existing test suite. Optionally you can configure, if you want to get 'actual result' and 'bug ID' of source test suite for retesting bug fix more efficiently.
-profile <user,password> [-file <file>]	, separated list of values, where the user and password must be specified. The user profile is exported to standard out

	or to an XML-file (parameter –file).
-results <xml-file>	updates the results of test cases which are described in the xml-file. For the xml-file the DTD has to be used, see 13.3.
-report < project,test structure,[version],[test suite],[filter],report setting,file>	, separated list of values, where the names for project, test structure, version, test suite name, filter, report setting, file must be specified and TEMPPO is opened with the specified version of the test structure (and optional a filter and test suite) and writes a report based on a report setting to the specified file. Remark: If there are a personal and project setting with the same name, the personal one is taken. If there are a personal and project filter with the same name, the personal one is taken. The file is specified only by name. The extension is specified by report type.
-file <file>	specifies a file (path + name). Only considered in conjunction with option "profile".
-exit	closes TEMPPO after test execution. Only considered in conjunction with option "run"
-help	print this message
-hide	hides TEMPPO during test execution. Only considered in conjunction with option "run"
-license	print information for license request

Use the batch file TEMPPO.bat in the installation directory for running TEMPPO with command line options:

```
TEMPPO.bat –export_teststructure "my project,the test
structure,V12,export.xml"
```

9 Plug-ins

TEMPPO offers a plug-in mechanism which automatically loads the implemented Java interface packed into a jar file. Each plug-in must be deployed in its own jar file and must be installed in the directory `<TEMPPO installation directoy>\plugins`.

A jar file always contains a manifest file which must contain at least the property `Plugin-Class`, which must provide the fully qualified name of the class implementing the plug-in. It's needed for loading the class.

Example for content of a manifest file:

```
Manifest-Version: 1.0
Plugin-Name: My First Plugin
Plugin-Version: 0.1
Plugin-Author: Muster Max
Plugin-Class: temppo.plugin.api.MyPlugin
```

9.1 Plug-in: Version Change Listener

A TEMPPO plug-in must implement the `Plugin` interface or one of its sub-interfaces.

`Plugin` is the super-interface for all plug-ins. It serves methods for basic plug-in information and load and unload handling (e.g. loading and saving plug-in settings).

`VersionListenerPlugin` is the interface for a concrete plug-in which can be used. It extends `Plugin` and serves as a hook for version change notification.

The plug-in developer has to implement a class-file packed into a jar file which extends `VersionListenerPlugin`. This class-file has to be implemented in the package `temppo.plugin.api`.

After starting TEMPPO the listener is registered and the actions of the plug-in are executed when the version is changed in TEMPPO (e.g. by activating another version of test structure, by creating a new test suite …).

10 Literature

/1/ Installation Guide
TEMPPO V6.0 installation and startup guide
March 2012

/2/ Glenford J. Myers
The Art of Software testing
USA: John Wiley & Sons, Inc. 1979
ISBN: 0-471-04328-1

11 Terms

Administration unit
This term denotes any object/data that is managed in a configuration management tool. Typical administration units are files or directories.

Branch
A linear sequence of versions is called a branch. It is possible to split off the main branch and create a new sequence of versions, independent from the main branch.

Check In / Out
As soon as an administration unit has been entered into the CM tool, it may only be changed, if a new version is created for it. The act of making it editable is called checking out. After an administration unit is checked out, any editing operations are allowed for it.
A new version is created when the unit is checked in. No further edition on this version is possible.

CM
Configuration Management: "Configuration is the process of identifying and defining the items in the system, controlling the change of these items throughout their lifecycle, recording and reporting the status of items and change requests, and verifying the completeness and correctness of items" [IEEE Std-729-1983]

Label
In virtually any CM tool, it is possible to attach a name to a version number. By doing so, it is possible to make the version history of administration units usable. Even after many successive versions, these important states can be easily found and restored by means of their descriptive label.

Main line / Main branch
The linear sequence of versions starting at the initial version.

Merge
Merging from version V1 to V2 means to take over changes in V1 to V2.

Test Case
„A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement." [IEEE Std-610.12-1990]

Test Case Design
Constructing test cases according to special methods (e.g. equivalence class) in a way such that from the description of the test package and from the requirement specification test cases can be derived independently.

Test Category
Category

Test Level
Level of the test: e.g. system test, acceptance test.

Test Package
Set of test packages and test cases.

Test Step
Single instruction for testing (usually with expected result).

Test Structure
The entire tree of test packages and test cases.

Test Suite
A subset of test cases from the test structure, intended for execution in a certain test + their recorded test results.

Version
A version denotes a certain state of an administration unit at a certain time.
There are two kinds of versions: (i) checked in version, which can't be changed any more, and (ii) checked out versions.
Usually, versions follow a decimal notation scheme, starting with "0" or "1". For each new version, the version number is incremented.

Version Tree
A tree representation of all versions (including all branches) of an administration unit.

12 Abbreviations

DB	Database
HW	Hardware
TD	Test Director
GUI	Graphical User Interface
SW	Software
DSN	Data Source Name
ODBC	Open Database Connectivity
UAI	Universal Automation Interface
CLI	Command Line Interface

13 Appendix

13.1 DTD-Schema for XML-Import (Test Structure)

```
<!ELEMENT TEMPPO_EXCHANGE ( PROJECT | TS | TP* ) >

<!-- NOTE: these attributes will not be part of the TEMPPO exchange format:
        creator CDATA #IMPLIED   - too many error possibilities, creator will be importer
        created CDATA #IMPLIED   - date will be taken from import
     NOTE: following elements we currently ignore
          PROJECT
         TS
-->

<!-- Common PCDATA elements -->
<!ELEMENT DESCRIPTION ( #PCDATA )>
<!ELEMENT PRECONDITION ( #PCDATA )>
<!ELEMENT POSTCONDITION ( #PCDATA )>
<!ELEMENT TESTGOAL ( #PCDATA )>
<!ELEMENT INSTRUCTION ( #PCDATA )>
<!ELEMENT INPUT ( #PCDATA )>
<!ELEMENT EXPECTED ( #PCDATA )>
<!ELEMENT VALUE ( #PCDATA )>

<!ELEMENT PROJECT ( DESCRIPTION, REQUIREMENTSTRUCTURE*, TS* ) >              <!-- currently
IGNORED -->
        <!ATTLIST PROJECT      name CDATA #REQUIRED >  <!-- the TEMPPO project must already
exist !! -->

<!ELEMENT TS ( DESCRIPTION, TP* ) >                                  <!-- currently IGNORED --
>
        <!ATTLIST TS   name CDATA #REQUIRED >
        <!ATTLIST TS   test_level CDATA #IMPLIED >    <!-- if it does not exist, autom. create
it -->
        <!ATTLIST TS   version CDATA #IMPLIED >       <!-- Added 31.5.2004 -->


<!ELEMENT TP (DESCRIPTION, PRECONDITION, POSTCONDITION, REQUIREMENT*, ( TP | TC )*) >
        <!ATTLIST TP   name CDATA #REQUIRED >        <!-- if zero-length then name "New
Testpackage" generated --><!-- if it already exists, add "_1", "_2", ... -->
        <!ATTLIST TP   test_category CDATA #IMPLIED > <!-- if it does not exist, autom. create
it for the test level of the containing test structure -->
        <!ATTLIST TP   document CDATA #IMPLIED >
        <!ATTLIST TP   doc_name CDATA #IMPLIED >
        <!ATTLIST TP   chapter CDATA #IMPLIED >
        <!ATTLIST TP   inherit_requirements (true | false) "false" >

<!ELEMENT TC ( TESTGOAL, PRECONDITION, POSTCONDITION, REQUIREMENT*, STEP*, AUTOMATION? ,
ADD_TEST_LEVEL*, USER_DEFINED_ATTRIBUTE*) >
        <!ATTLIST TC   user_defined_id CDATA #IMPLIED >  <!-- if it already exists in
containing test structure, add "_1", "_2!, ... -->
        <!ATTLIST TC   name CDATA #REQUIRED >           <!-- if zero-length then name "New
Testcase" generated --><!-- if it already exists in containing test package, add "_1", "_2!,
... -->
  <!ATTLIST TC situation (Error | Regular) "Error" >
        <!ATTLIST TC   val_ver (Validation | Verification) "Validation" >
        <!ATTLIST TC   priority (Top | High | Medium | Low) "Top" >
        <!ATTLIST TC   state (Designed | In_Work | Ready_for_Review | Ready | Approved)
"Designed" >
        <!ATTLIST TC   inherit_requirements (true | false) "false" >
```

```
<!ELEMENT REQUIREMENT (DESCRIPTION, REQUIREMENTSTRUCTURE) >
       <!ATTLIST REQUIREMENT  id        CDATA #REQUIRED > <!-- cannot be zero-length string--
><!-- if it does not exist, it will be automatically created -->

<!ELEMENT REQUIREMENTSTRUCTURE (DESCRIPTION) >
       <!ATTLIST REQUIREMENTSTRUCTURE       name        CDATA #REQUIRED > <!-- cannot be
zero-length string-->
       <!ATTLIST REQUIREMENTSTRUCTURE       version     CDATA #REQUIRED > <!-- cannot be
zero-length string-->

<!ELEMENT STEP (INSTRUCTION, INPUT, EXPECTED, USR_FIELD*) >

<!ELEMENT USR_FIELD (VALUE)>
       <!ATTLIST USR_FIELD    name CDATA #REQUIRED >
       <!ATTLIST USR_FIELD    type (text | image) "text" >

<!ELEMENT AUTOMATION EMPTY >
       <!ATTLIST AUTOMATION   tool ( SilkTest | WinRunner | UniversalFile | UniversalSocket |
JUnit | QuickTestPro | TestPartner) "UniversalFile"  >
       <!ATTLIST AUTOMATION   file CDATA #IMPLIED >
       <!ATTLIST AUTOMATION   data CDATA #IMPLIED >

<!ELEMENT ADD_TEST_LEVEL EMPTY >
       <!ATTLIST ADD_TEST_LEVEL      name CDATA #REQUIRED > <!-- cannot be zero-length
string--><!--if it does not exist, it will be automatically created -->

<!ELEMENT USER_DEFINED_ATTRIBUTE EMPTY >
       <!ATTLIST USER_DEFINED_ATTRIBUTE      name  CDATA #REQUIRED >      <!-- cannot be zero-
length string--><!-- if it does not exist, it will be automatically created -->
       <!ATTLIST USER_DEFINED_ATTRIBUTE      value CDATA #REQUIRED >      <!-- cannot be zero-
length string--><!-- if it does not exist, it will be automatically created -->
```

13.2 DTD-Schema for XML-Import (Test Suite)

```
<!ELEMENT TEMPPO_EXCHANGE ( PROJECT, TS, EXPORT_ID, TEST_SUITE ) >

<!-- Common PCDATA elements -->
<!ELEMENT DESCRIPTION ( #PCDATA )>
<!ELEMENT PRECONDITION ( #PCDATA )>
<!ELEMENT POSTCONDITION ( #PCDATA )>
<!ELEMENT TESTGOAL ( #PCDATA )>
<!ELEMENT RESULTOUTPUT ( #PCDATA )>
<!ELEMENT INSTRUCTION ( #PCDATA )>
<!ELEMENT INPUT ( #PCDATA )>
<!ELEMENT EXPECTED ( #PCDATA )>
<!ELEMENT ACTUALRESULT ( #PCDATA )>
<!ELEMENT BUGID ( #PCDATA )>
<!ELEMENT VALUE ( #PCDATA )>


<!ELEMENT PROJECT (DESCRIPTION)>
       <!ATTLIST PROJECT      name CDATA #REQUIRED >

<!ELEMENT TS (DESCRIPTION) >
       <!ATTLIST TS   name CDATA #REQUIRED >
       <!ATTLIST TS   test_level CDATA #IMPLIED >    <!-- if it does not exist, autom. create
it -->
       <!ATTLIST TS   version CDATA #IMPLIED >
```

```
        <!ATTLIST TS   lastupdate CDATA #IMPLIED >
        <!ATTLIST TS   lastupdateby CDATA #IMPLIED >

<!ELEMENT EXPORT_ID EMPTY>
        <!ATTLIST EXPORT_ID   value CDATA #REQUIRED >

<!ELEMENT TEST_SUITE ( DESCRIPTION, TP*, USER_DEFINED_ATTRIBUTE_TSUITE* ) >
        <!ATTLIST TEST_SUITE  id CDATA #REQUIRED >
        <!ATTLIST TEST_SUITE  name CDATA #REQUIRED >
        <!ATTLIST TEST_SUITE  lastupdate CDATA #IMPLIED >
        <!ATTLIST TEST_SUITE  lastupdateby CDATA #IMPLIED >

<!ELEMENT TP (DESCRIPTION, PRECONDITION, POSTCONDITION, REQUIREMENT*, ( TP | TEST_RESULT )*) >
        <!ATTLIST TP   name CDATA #REQUIRED >         <!-- if zero-length then name "New
Testpackage" generated --><!-- if it already exists, add "_1", "_2", ... -->
        <!ATTLIST TP   test_category CDATA #IMPLIED > <!-- if it does not exist, autom. create
it for the test level of the containing test structure -->
        <!ATTLIST TP   document CDATA #IMPLIED >
        <!ATTLIST TP   doc_name CDATA #IMPLIED >
        <!ATTLIST TP   chapter CDATA #IMPLIED >
        <!ATTLIST TP   inherit_requirements (true | false) "false" >
        <!ATTLIST TP   lastupdate CDATA #IMPLIED >
        <!ATTLIST TP   lastupdateby CDATA #IMPLIED >


<!ELEMENT TEST_RESULT ( TESTGOAL, PRECONDITION, POSTCONDITION, RESULTOUTPUT, REQUIREMENT*,
TEST_STEP_RESULT*, AUTOMATION? , ADD_TEST_LEVEL*, USER_DEFINED_ATTRIBUTE*,
USER_DEFINED_ATTRIBUTE_TRESULT*) >
        <!ATTLIST TEST_RESULT   tc_id CDATA #REQUIRED>
        <!ATTLIST TEST_RESULT  user_defined_id CDATA #IMPLIED >  <!-- if it already exists in
containing test structure, add "_1", "_2!, ... -->
        <!ATTLIST TEST_RESULT  name CDATA #REQUIRED >            <!-- if zero-length then name
"New Testcase" generated --><!-- if it already exists in containing test package, add "_1",
"_2!, ... -->
  <!ATTLIST TEST_RESULT       type (Manual | Automated) "Manual">
        <!ATTLIST TEST_RESULT  tester CDATA #REQUIRED >
        <!ATTLIST TEST_RESULT  tested CDATA #REQUIRED >
        <!ATTLIST TEST_RESULT  situation (Error | Regular) "Error" >
        <!ATTLIST TEST_RESULT  val_ver (Validation | Verification) "Validation" >
        <!ATTLIST TEST_RESULT  priority (Top | High | Medium | Low) "Top" >
        <!ATTLIST TEST_RESULT  state (Designed | In_Work | Ready_for_Review | Ready | Approved)
"Designed" >
        <!ATTLIST TEST_RESULT inherit_requirements (true | false) "false" >
        <!ATTLIST TEST_RESULT result ( Failed | Passed | Not_executed | Blocked |
Not_completed | Not_implemented | On_hold) "Not_executed" >
        <!ATTLIST TEST_RESULT  lastupdate CDATA #IMPLIED >
        <!ATTLIST TEST_RESULT  lastupdateby CDATA #IMPLIED >


<!ELEMENT REQUIREMENT (DESCRIPTION) >
        <!ATTLIST REQUIREMENT  REQUIREMENTID        CDATA #REQUIRED > <!-- cannot be zero-
length string--><!-- if it does not exist, it will be automatically created -->


<!ELEMENT TEST_STEP_RESULT (INSTRUCTION, INPUT, EXPECTED, ACTUALRESULT, BUGID, USR_FIELD*) >
        <!ATTLIST TEST_STEP_RESULT    test_step_id  CDATA #REQUIRED >
        <!ATTLIST TEST_STEP_RESULT    result        ( Failed | Passed | Not_executed | Blocked
| Not_implemented | On_hold) "Not_executed"  >


<!ELEMENT USR_FIELD (VALUE)>
        <!ATTLIST USR_FIELD   name CDATA #REQUIRED >


<!ELEMENT AUTOMATION EMPTY >
        <!ATTLIST AUTOMATION  tool ( SilkTest | WinRunner | UniversalFile | UniversalSocket |
JUnit ) "UniversalFile"  >
        <!ATTLIST AUTOMATION  file CDATA #IMPLIED >
        <!ATTLIST AUTOMATION  data CDATA #IMPLIED >


<!ELEMENT ADD_TEST_LEVEL EMPTY >
        <!ATTLIST ADD_TEST_LEVEL      name CDATA #REQUIRED > <!-- cannot be zero-length
string--><!--if it does not exist, it will be automatically created -->
```

```
<!ELEMENT USER_DEFINED_ATTRIBUTE EMPTY >
       <!ATTLIST USER_DEFINED_ATTRIBUTE      name  CDATA #REQUIRED >      <!-- cannot be zero-
length string--><!-- if it does not exist, it will be automatically created -->
       <!ATTLIST USER_DEFINED_ATTRIBUTE      value CDATA #REQUIRED >      <!-- cannot be zero-
length string--><!-- if it does not exist, it will be automatically created -->


<!ELEMENT USER_DEFINED_ATTRIBUTE_TRESULT EMPTY >
       <!ATTLIST USER_DEFINED_ATTRIBUTE_TRESULT     name  CDATA #REQUIRED >      <!-- cannot
be zero-length string--><!-- if it does not exist, it will be automatically created -->
       <!ATTLIST USER_DEFINED_ATTRIBUTE_TRESULT     value CDATA #REQUIRED >      <!-- cannot
be zero-length string--><!-- if it does not exist, it will be automatically created -->


<!ELEMENT USER_DEFINED_ATTRIBUTE_TSUITE EMPTY >
       <!ATTLIST USER_DEFINED_ATTRIBUTE_TSUITE      name  CDATA #REQUIRED >      <!-- cannot
be zero-length string--><!-- if it does not exist, it will be automatically created -->
       <!ATTLIST USER_DEFINED_ATTRIBUTE_TSUITE      value CDATA #REQUIRED >      <!-- cannot
be zero-length string--><!-- if it does not exist, it will be automatically created -->
```

13.3 DTD-Schema for XML-Import (Results) with CLI

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT TEMPPO_EXCHANGE ( LOGIN, RESULTS ) >

<!ELEMENT USER ( #PCDATA )>
<!ELEMENT PASSWORD ( #PCDATA )>
<!ELEMENT PROJECT ( #PCDATA )>
<!ELEMENT LOGIN ( USER, PASSWORD, PROJECT )>

<!ELEMENT SUMMARY ( #PCDATA )>
<!ELEMENT RESULT ( SUMMARY? )>
       <!ATTLIST RESULT
              testsuiteid CDATA #REQUIRED
              testcaseid CDATA #REQUIRED
              value CDATA #REQUIRED
       >
<!ELEMENT RESULTS ( RESULT* )>
```

13.4 DTD-Schema for XML-Export (User Profile) with CLI

```
<!ELEMENT LOGIN ( ROLES?, BUGZILLA? )>
       <!ATTLIST LOGIN
              name CDATA #REQUIRED
              superuser CDATA #IMPLIED
              result CDATA #REQUIRED
       >

<!ELEMENT PROJECT ( #PCDATA )>
<!ELEMENT ROLE ( PROJECT* )>
       <!ATTLIST ROLE
              name CDATA #REQUIRED
       >
<!ELEMENT ROLES ( ROLE* )>

<!ELEMENT EMAIL ( #PCDATA )>
<!ELEMENT PASSWORD ( #PCDATA ) >
<!ELEMENT BUGZILLA (EMAIL+, PASSWORD+)>
```

13.5 Predefined Functions

Cannot be changed.

13.6 Predefined Roles

The following roles are predefined in TEMPPO:

- TEMPPO Super user: All functions activated
- TEMPPO Key User: Most important or used functions activated
- TEMPPO Key Tester: Important function concerning Test Suite
- TEMPPO Tester: Most important functions concerning Test Suite
- TEMPPO Requirement Editor: All functions in Requirement Manager plus all functions concerning requirements in TEMPPO and TEMPPO Administrator
- TEMPPO Editor :All functions for creating Test Packages and Test Cases
- TEMPPO Reporter :All functions for creating reports and analysis
- TEMPPO Guest: All functions for viewing and exporting data

Those roles can be changed – except TEMPPO Super user.

13.6.1Functions to be applied to a role

An “X” means that it can be (de)activated for a role.

		Right						
Application	**Function**	**New / Execute**	**Change**	**Delete**	**Read**	**Move ownership**	**Consider user**	**Comment**
TEMPPO Administrator	Administrate locks	X						
	Administrate projects	X	X	X	X			
	Administrate roles	X	X	X	X	X		Change includes assignment of roles to users
	Administrate test structures				X			
	Administrate test suites				X		X	
	Assign Requirements	X		X				
	Assign Test category	X		X				
	Assign Test level	X		X				
	Assign UDA	X		X				
	Assign Uploads	X		X				
	Assign User Fields	X		X				
	Consistency checker	X						
	Freeze Test Structure	X						
	Freeze Test Suite	X						
	Move ownership	X						
	Unfreeze Test Structure	X						
	Unfreeze Test Suite	X						
	Unlock test suites	X						
Administrator Metadata	Requirements	X	X	X	X	X	X	
	Test categories	X	X	X	X	X	X	

	Test levels	X	X	X	X	X	X	
	Uploads	X	X	X	X	X	X	
	User fields	X	X	X	X	X	X	
	User defined attributes	X	X	X	X	X	X	
TEMPPO Test structure	Apply requirement updates	X						
	Edit	X	X			X		Read test structure means any version, not only main/LATEST
	Export	X						
	Import	X						
	JUnit Interface	X						
	Unlock items	X					X	Test structure root, TP, TC
TEMPPO Test package	Assign requirements	X		X				
	Assign test categories	X						
	Edit	X	X	X		X	X	
TEMPPO Test case	Assign requirements	X		X				Consider User: deleting requirement assignment is only allowed if user is TC editor
	Assign test level	X		X				
	Assign test case attributes	X		X				
	Edit	X	X	X		X	X	Consider User concerns only the deleting
	Merge	X					X	Consider User: merging to main/LATEST is only allowed if user is editor of main/LATEST version
TEMPPO Test suite	Assign test suite attributes	X		X				
	Edit	X	X			X		Delete is only allowed, if test suite has no test results
	Export	X						
	Export To Doors	X						
	Import	X						
	Remove from export database	X						
	Unlock exported test suite	X					X	
	Unlock items	X					X	
TEMPPO Test result	Assign test result attributes	X		X				
	Edit	X	X					Implies “Run automated” and “Set results (multi)”

	Merge actual execution time	X						
TEMPPO Evaluation	Analyse	X						
	Analyse by bug ID	X						
	Creation progress	X						
	Execution progress	X						
	Personal report setting	X	X	X	X	X		
	Project report setting	X	X	X	X	X		
	Report	X						
	Requirement analysis	X						
TEMPPO Requirement Manager	Export to XML	X						
	Import from CSV	X						
	Import from RM Tool	X						
	Import from XML	X						
	Open	X						
	Requirement	X	X	X	X			
	Requirement Structure	X	X					
	Split	X						
	Update	X						
	Versions	X						
TEMPPO Others	Multi Selection	X						
	Copy	X						
	Copy All	X						
	Export / Import filter	X						
	Filter	X	X	X	X	X		New implies applying a filter
	Order	X					X	
	Task list	X	X	X	X			
	Versions	X	X	X				

13.7 UAI Adapters

An adapter is used to convert the results returned from an automation tool into a format, which can be processed by TEMPPO. Currently 2 adapters are provided.

13.7.1CDefaultAutoToolAdapter

The result string returned after a test case has been executed must have following format:

```
line 1      result=(passed|failed|blocked|not implemented|on hold)
line 2      summary_begin
            ....
            ....
            ....
line n      summary_end
```

13.7.2CAPOXIAdapter

Taken from APOXI user’s guide:

“After finishing the test, an exit code is returned to the client, which is 1 if the test was successful, 2 if the test failed and 3 if the test was not completed. Additionally, the client gets also a string, which contains the detailed test result (passed/failed, number of errors and warnings).”

13.7.3How to implement an adapter

13.7.3.1 Classes

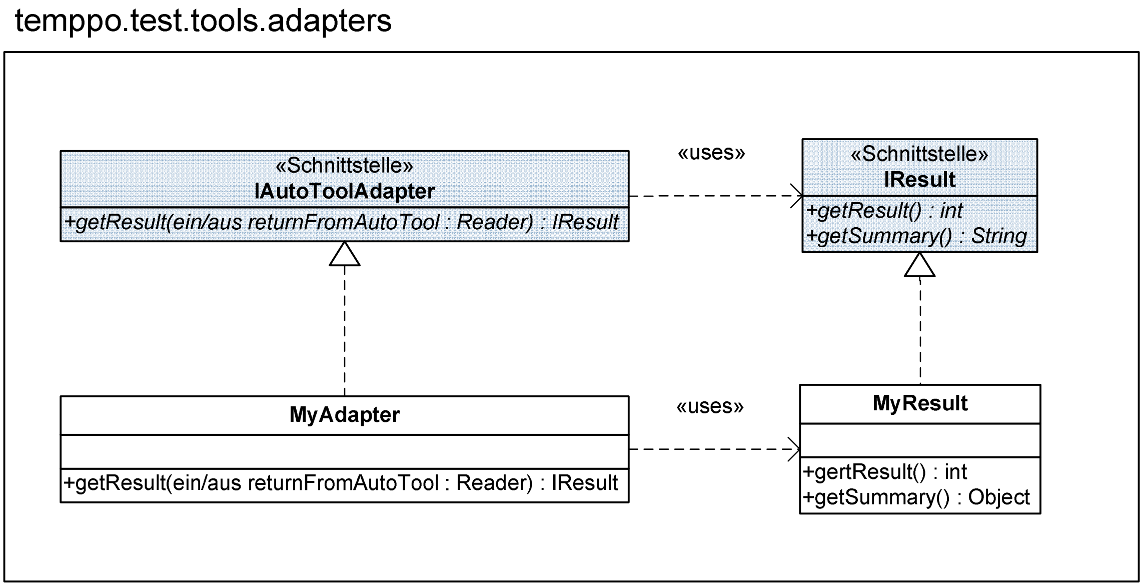


Figure 381: Class diagram of an adapter implementation

Interface: *temppo.test.tools.adapters.IAutoToolAdapter*

An adapter must implement this interface.

public IResult getResult(Reader returnedFromAutoTool) throws Exception

This method is called during test execution for each test case. Via `returnedFromAutoTool` the adapter must extract the result and return it as an `IResult` implementation.

Interface: *temppo.test.tools.IResult*

The result returned by the adapter must implement this interface.

`public int getResult()`

Must return one of following results (defined in IResult):
PASSED, FAILED, BLOCKED, NOT_EXECUTED, NOT_COMPLETED, NOT_IMPLEMENTED, ON_HOLD

`public String getSummary`

Must return the summary for the test result.

13.7.3.2 Code Snippet

The code snippet below can be used as body for an adapter implementation. It has to extended by parsing the result handed over from the automation tool:

```
package ext;

import java.io.Reader;

import temppo.test.tools.adapters.IAutoToolAdapter;
import temppo.test.tools.adapters.IResult;

/**
 * Implementation of adapter for universial automation interface.
 */
public class MyAdapter implements IAutoToolAdapter {

  public IResult getResult(Reader returnFromAutoTool) throws Exception {
    Result ret = new Result();

    /*

    TODO: add parsing of returnFromAutoTool here and set result and summary

    Example:

    // use buffered reader to process returnFromAutoTool line per line
    BufferedReader br = new BufferedReader(returnFromAutoTool);
    // read all lines and search for result and summary
    String line = null;
    StringBuffer summary = new StringBuffer();
    while ((line = br.readLine()) != null) {
      if (line.startsWith("Result: ")) {
        // process result here
        if (line.equals("Result: FAILED")) {
          ret.result = IResult.FAILED;
        } else if (line.equals("Result: PASSED")) {
          ret.result = IResult.PASSED;
        }
      } else {
        // rest is for summary
        summary.append(line);
        summary.append(System.getProperty("line.separator"));
      }
    }
    ret.summary = summary.toString();
    */

    return ret;
  }

  /**
   * Implementation of result interface.
   */
  class Result implements IResult {

    String summary = "";

    int result = BLOCKED;

    public String getSummary() {
      return summary;
    }

    public int getResult() {
      return result;
    }
  }

}
```

13.7.3.3 Implementation and integration steps

1. Create following subdirectories in your TEMPPO installation directory

   ```
   %INSTDIR%\dev\src\ext
   %INSTDIR%\dev\bin
   ```

2. Create the file %INSTDIR%\dev\src\ext\MyAdapter.java and implement the adapter.
3. For compilation a JDK 1.5 is needed. Execute following command in %INSTDIR%\dev:

   ```
   javac -classpath ..\lib\temppo.jar -d bin src\ext\MyAdapter.java
   ```

4. Pack the adapter into a jar file. Execute following command in `%INSTDIR%\dev`:

   ```
   jar -cf adapter.jar -C bin ext
   ```

5. Add adapter.jar to the classpath for TEMPPO and TEMPPO Administrator via extending `common.classpath` in `%INSTDIR%\launcher.properties`:

   ```
   lib\\temppo.jar;lib\\lib.jar;lib\\commons-codec.jar;\
   lib\\commons-httpclient.jar;lib\\commons-logging.jar;\
   lib\\commons-cli.jar;lib\\junit.jar;lib\\registry.jar;\
   lib\\sqljdbc.jar;lib\\oracle.jar;lib\\pgjdbc.jar;\
   ```

```
lib\\jcommon.jar;lib\\jfreechart.jar;dev\\adapter.jar
```

6. The adapter is now available in TEMPPO Administrator for Universal
7. Socket configuration.
8. For configuration of UniversalFile, admin.properties has to be adapted:

```
automation.UniversalFile.adapter=ext.MyAdapter
```

14 Index