TETware *professional*

User Guide

for TETware *professional* 1.4

Released: 10 January 2002

THE Open GROUP

The information contained within this document is subject to change without notice.

This document is produced by

The Open Group
Apex Plaza
Forbury Road
Reading
Berkshire
RG1 1AX
England.

CONTENTS

CONTENTS

CONTENTS

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Preface

This document is the TETware *professional* User Guide.

TETware *professional* is a Test Execution Management System that takes care of the administration, reporting, and sequencing of the tests providing a single common user interface for all of the tests that you develop.

TETware *professional* has been tested and used on UNIX, Linux and Windows operating systems. It includes all of the functionality of TETware together with a number of new features.

Throughout this document, the Windows NT, 2000 and 9x operating systems are referred to collectively as **Win32** systems. The individual names are only used when it is necessary to distinguish between them.

1.2 Audience

This document is intended to be read by systems administrators who will install TETware *professional* on their computer systems, and by software testing engineers who will use TETware *professional* to run verification test suites.

1.3 Conventions Used in this Guide

The following typographic conventions are used throughout this guide:

- `Courier font` is used for function and program names, literals and file names. Examples and computer-generated output are also presented in this font.

- The names of variables are presented in *italic font*. You should substitute the variable's value when typing a command that contains a word in this font.

- **Bold font** is used for headings and for emphasis.

1.4 Related Documents

Refer to the following documents for additional information about TETware *professional*:

TETware *professional* Installation Guide
TETware *professional* Demonstration Guide
TETware *professional* Release Notes
TETware *professional* Help Pages
TETware *professional* Real Time and Embedded Systems Extension

The TETware *professional* Installation Guide contains important information about how to install and use TETware *professional*. You should read the Installation Guide thoroughly before attempting to install and use each new release of TETware *professional.*

1.5 Problem Reporting

If you have subscribed to TETware *professional* support and you encounter a problem while installing and using TETware *professional*, you can send a support request by electronic mail using the dedicated email address that is provided. Evaluators should email to tetware_manager@opengroup.org

All Problem Reports are welcome and actively encouraged. The more problems that are found and fixed the better the product will be. Please submit all bugs and queries found. Also, please submit requests for features and upgrades.

2 TETware *professional* Overview

2.1 Architecture

TETware *professional* is written as a wrapper around a core. The wrapper is written purely in Java 2 using the full capabilities of the Swing APIs. Please note that TETware *professional* requires Java 2 (also known as JDK 1.2) or later.

The core is written in C, and requires a C compiler.

TETware *professional* requires POSIX.1 conformance. It is supported on UNIX operating systems, Linux and 32-bit Windows Systems.

The system requirements for Windows Systems are:

- Visual C++ 4.0 or later
- MKS Toolkit for Windows NT Version 4.4 or later

Language bindings are available for the development of test cases in: C, C++, Java, Perl, Korn Shell, XPG3 Shell, and Tcl.

2.2 Test Run

TETware *professional* uses the concept of a `Test Run`; this is a logical and conceptual unit that encompasses all the necessary data needed to ensure that a series of Test Cases can be executed successfully.

The `Test Run` contains a number of elements, some mandatory and others optional that when combined enable the User to concentrate more on the purpose behind the tests.

The `Test Run` contains configuration data for both local and, optionally, remote machines involved in testing. It includes a `Scenario` file that will be used to manipulate the tests. It also contains information relating to the operation of TETware *professional* including the advanced features that TETware *professional* offers.

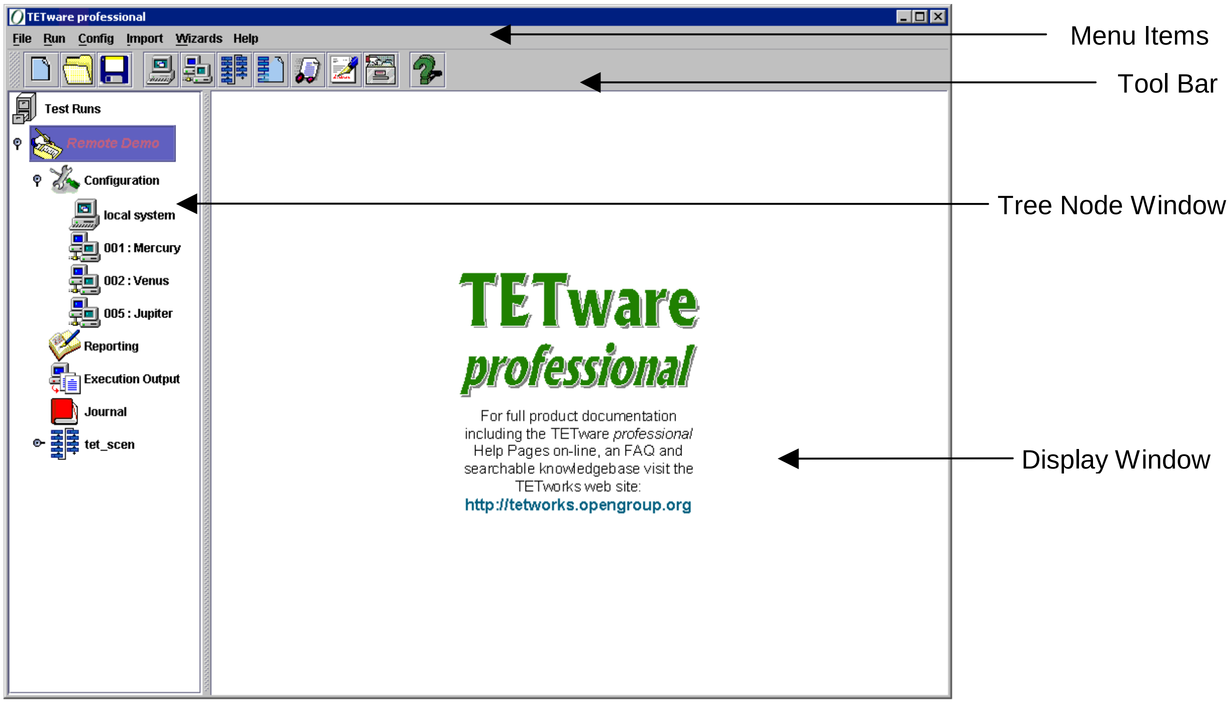
Because the Test Run is a logical unit, it can be saved as one unit, meaning that all the data relating to a given execution of tests is saved as a whole. This means that the User no longer has to worry about the management of configuration files and environment variables.

2.3 Scenario File

A `scenario` file contains one or more test scenarios for a test suite. Each test scenario lists all the test cases and describes how they are to be processed.

When a Test Run is created, there will be no Scenario file included. This is shown at the bottom of the Test Run tree with the label: No Scenario. The user MUST add an existing scenario file, or create a new scenario file in order to run the Test Run. However, once that Test Run is saved, that scenario file is saved with the Test Run.

2.4 The GUI



For full product documentation including the TETware *professional* Help Pages on-line, an FAQ and searchable knowledgebase visit the TETworks web site: http://tetworks.opengroup.org

Figure 1: TETware *professional* GUI

2.4.1 Tree Node Window

The Tree Node Window is where the Test Runs are listed. Since a number of Test Runs can be managed at any given time, the concept of an ‘Active Test Run’ is introduced. The Active Test Run is the Test Run that is currently being examined or manipulated by the User. The configuration screens, reporting node, execution node, journal node and scenario node will all display information relating to the Active Test Run.

The Active Test Run is indicated in TETware *professional* by the Test Run name being highlighted in **RED**.

You can select a Test Run to be ‘Active’ by clicking on the Test Run itself or any of it's child nodes, so by clicking on the Scenario file you would make the Test Run associated with that Scenario file the Active Test Run. Once a selection is made, you can move up or down the tree by pressing the up or down arrow keys. Any node clicked on in TETware *professional* will be highlighted in **BLUE**.

Moving between Test Runs will cause the associated screens to change, dependent on the data associated with the Test Run chosen. At any given time the configuration screens will display the Active Test Runs configuration data.

Please note that expanding and contracting the tree nodes via the use of the folder handles will NOT cause the Active Test Run or selected node to change.

2.4.2 The Tree Nodes

Each Test Run is represented in the Test Run tree by a series of tree nodes, each of which give the User a ‘window’ into a different aspect of the Test Run and it's data. Right clicking on most of these nodes provides the user with access to Test Run configuration options.

The main ‘root’ node that all Test Runs are added to is represented by the icon 🗄 .

When a Test Run is created or opened this node 🗄 is also opened.

The following table gives a description of each tree node and it's usage within the Test Run.

Tree Node Icon	Tree Node Name	Description
	Test Run Open	This node is the base node for all the subsequent nodes relating to an individual Test Run, from this node you are able to add a scenario file, save and remove the Test Run. It is the starting point for the Test Run.
	Test Run Closed	This node is the closed version of the Test Run Open node described above; when this icon is displayed it indicates that the Test Run has been hidden in the tree.
	Configuration	This node is starting point for the configuration for the Test Run. Attached to this node is a 'local system' node, which holds the local systems configuration. Remote systems are also attached to this node. If you click on this node then the Main configuration screen for the Test Run will be displayed.
🖥	Local System	This node represents the local system configuration. The local system is the system on which TETware *professional* will be invoked. Clicking on this node displays the local system configuration screens.
	Remote System	This node represents a remote system that will be used in the Test Run. This node is optional and is only required when TETware *professional* is being used in distributed mode, i.e. for remote or distributed testing. Clicking on this node displays the configuration screens for that remote system.
	Reporting	Clicking on this node will take you to the Reporting screen where you can specify a report to be generated using a Journal file or configure the database extract tool and specify the Journal file parameters to be passed to the database.
⛔	Execution Output Started	As the name suggests this node, when shown, indicates that the Test Run is currently executing. The output can be viewed in the Execution screens. The Execution screens can be seen by clicking on this node. This icon will be displayed for as long as the Test Run is executing.
	Execution Output Stopped	As the name suggests this node, when shown, indicates that the Test Run is NOT currently executing. Clicking on this icon will display the Execution screens.
📕	Journal	Clicking on this icon will take you to the Journal file display screen. If you have performed a Test Run for which there was a Journal file generated then it will be displayed here. Right clicking on this node allows you to Browse the file system for a Journal file. *Note: you cannot open a Journal file that is greater than 2MB in size.*
	Scenario File	This node represents the Scenario file. If valid scenarios exist within the Scenario file then these are displayed as scenario nodes below this node. Clicking on this node will display the Scenario file in the Scenario file display window. Right clicking on this node allows you to refresh the scenario file (basically re-read the Scenario file if it has changed), include and exclude all the scenarios, remove the Scenario file, and display the path to the Scenario file. If no scenario file is included then right clicking on this node allows you to add an existing scenario file, or create a new scenario file.
✓	Scenario Included	This node represents a scenario from the Scenario file that should be Included when the Test Run is executed. Single clicking on the scenario node will toggle its included/excluded status.
✗	Scenario Excluded	This node represents a scenario from the Scenario file that should be Excluded when the Test Run is executed. Single clicking on the scenario node will toggle its included/excluded status

Table 1: Tree Node Icons

2.4.3 The Tool Bar

The Tool Bar, shown below, is provided to allow the User access to certain frequently used functions.

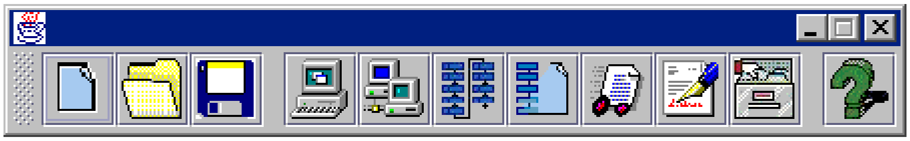


Figure 2: The Toolbar (undocked)

It is possible to ‘dock’ the toolbar into any of the edges of the GUI.

It is also possible to have the toolbar floating in it's own frame, as in Figure 2, this gives the User maximum flexibility as to the positioning of the toolbar. To undock the toolbar click on the area to the left of the New Test Run icon 🗋 and drag it to the new position while holding the left mouse button down.

2.4.4 Toolbar items

Table 2 describes the icons on the toolbar and their function:

Icon	Name	Description
🗋	New Test Run	Create a new blank Test Run, the User will be prompted for the Test Run name.
📂	Open an existing Test Run	This will display a file selection dialog that the User can browse to open an existing Test Run, a file filter of *.trf is provided to find Test Run files quickly and easily.
💾	Save the Active Test Run	This button will perform the same function as the ‘Save’ function in the ‘File’ menu. Note: this button will only work when there is an active Test Run.
🖥	Start New Test Run Wizard	This button starts the Test Run Setup (Basic) Wizard. See Section 12.1.
🖧	Start New Remote System Wizard	This button starts the New Remote System Wizard. See Section 12.3.
	Add a Scenario file	When a test run is active, this button opens a file chooser box and prompts the user to select a Scenario File. The selected file is then added to the active Test Run. If there is no active test run an error dialog box is displayed.
	Create a Scenario file	When a test run is active, this button takes the user to the Create Scenario File tab.
	Execute Test Run	This button will start the execution of the active Test Run. The user is prompted for confirmation before the Test Run executes.
📝	Run the TETware Report Writer	This button generates a Report using the TETware Report Writer using the Journal file specified in the TETware Report Writer tab.
🗄	Start Extract to Database	This button starts the process of extracting Journal file parameters and passing them to the database specified on the Database Setup tab.
?	Help	This button launches the TETware *professional* Help pages.

Table 2: Toolbar Icons

2.4.5 Menu Items

2.4.5.1 File Menu

Menu item	Description	See Section
New Test Run	Create a new Test Run.	3.1.2
Open Test Run	Open an existing Test Run.	3.1.1
Save Test Run	Save the currently Active Test Run.	11.1
Save As Test Run	Save the currently Active Test Run as a different name.	11.2
Remove Test Run	Remove the currently Active Test Run.	3.2
Exit	Exit TETware *professional*	14

2.4.5.2 Run Menu

Menu item	Description	See Section
Execute Test Run	Executes the currently Active Test Run.	8.2
Quick Re-run	Re-execute a Test Run for all Tests that have a result code of FAILED and/or UNTESTED but only for the Modes of Operation used for the initial execution.	8.3
Advanced Re-run	Re-execute a Test Run based upon the criteria specified by the User.	8.4
Stop Test Run	Stops a Test Run executing (if already executing).	8.5

2.4.5.3 Config Menu

Menu item	Description	See Section
Add Scenario File	Add a Scenario File to the currently Active Test Run (if one is not already present).	5.2
Remove Scenario File	Remove the Scenario File from the currently Active Test Run (if one is already present).	5.7
Create Scenario File	Create a Scenario File and add it to the currently Active Test Run	5.6
Add Remote System	Add a Remote System to the currently Active Test Run.	6.1
Save as default settings	Save the currently Active Test Runs main and local system configuration as the defaults.	11.3
Revert to default settings	Set the currently Active Test Runs main and local system configuration to the defaults.	11.3

2.4.5.4 Import Menu

Menu item	Description	See Section
Import Build mode file	Import a Build mode file into the currently Active Test Run.	6.3.1
Import Execute mode file	Import an Execute mode file into the currently Active Test Run.	6.3.2
Import Clean mode file	Import a Clean mode file into the currently Active Test Run.	6.3.3
Import Distributed config file	Import a Distributed config file into the currently Active Test Run.	4.3.2

2.4.5.5 Wizard Menu

Menu item	Description	See Section
Test Run Setup (Basic) Wizard	Steps the User through the four basic steps that must be carried out to configure a new Test Run	12.1
Test Run Setup (Advanced) Wizard	Steps the User through all the steps that might be carried out to configure a new Test Run.	12.2
Start New Remote System Wizard	Steps the User through all the steps that might be carried out to configure a new Remote System	12.3

2.4.5.6 Help Menu

Menu item	Description	See Section
About TETware *professional*	Display information relating to the TETware *professional*	(none)
Browse Help pages	Browse these help pages.	13.1

2.4.6 Keyboard Shortcuts

The menus and menu items can also be accessed by using keyboard shortcuts as follows:

Alt-F	File menu	Alt-E	Execute Test Run
Alt-N	New test Run	Alt-C	Configuration menu
Alt-O	Open Test Run	Alt-I	Import menu
Alt-S	Save Test Run	Alt-W	Wizard menu
Alt-A	Save As Test Run	Alt-H	Browse Help Pages
Alt-X	Exit	Alt-D	Database Extract
Alt-R	Run menu	Alt-G	Generate Report

2.5 Dialog Boxes

The following buttons are general to most of the dialog boxes and tabs found within the GUI:

`BROWSE` Clicking the `Browse` button will display a file selection box, which can be used to browse the file system to make selections.

`CLEAR` clear the field you want to enter a variable into or clear whatever you have previously entered

`CANCEL` aborts the present operation entirely.

`BACK` go back one step to the previous field.

`NEXT` move to the next field.

`FINISH` finish with the settings of the variable, save your settings and exit.

2.6 Modes of Operation

TETware *professional* operates in one of several modes as follows:

`build mode` in this mode, test cases are built or compiled

`execute mode` in this mode, Test Cases are run.

`clean modes` in this mode, test cases are cleaned; i.e., the test case directory is returned to the state that it was in before the test case was built and/or executed. e.g. all files created are deleted.

One or more modes of operation are specified by command-line options each time that TETware *professional* is invoked.

2.7 Local, Remote and Distributed Testing

The Lite version of TETware *professional* is able to process non-distributed test cases on the Local System.

The distributed version of TETware *professional* is able to process non-distributed test cases on both the Local System and on one or more Remote Systems. In addition, it is able to process distributed test cases where parts of each test case execute on the Local System and/or on one or more (up to 999) Remote Systems.

Distributed TETware *professional* systems are identified by a three-digit system ID. System ID `zero` (000) refers to the Local System and other system ID values refer to Remote Systems. In this context, the Local System is defined as the one on which TETware *professional* is invoked. Usually, different TETware *professional* systems are located on different physical machines. However, there is no reason why different system IDs should not refer to the same machine.

2.8 Embedded System Testing

TETware *professional* is able to control the execution of tests on an Embedded POSIX Realtime System that supports one of the profiles defined in POSIX 1003.13. For detailed information about Embedded Realtime System, please refer to the TETware *professional* Realtime and Embedded Systems Extension.

3 Getting Started

3.1 Opening Test Run File(s)

Once TETware *professional* has finished loading and the main screen is open, you are able to either:

- Open an existing Test Run.
- Create a New Test Run.

3.1.1 Opening an Existing Test Run File(s)

An existing Test Run File is one that has been created at some point in the past and may be in any state, i.e. the configuration may be set or there may be a scenario file added or a combination of the two.

There are a number of ways to open an existing Test Run:

- Use the icon in the toolbar 📁.
- Use the `Open Test Run` item in the File menu.
- Right click on the `Test Runs` tree node and choose `Open Test Run`.
- Use keyboard shortcut Alt-O.

Whichever method you choose you will then be presented with a file selection dialog box captioned `Open Test Run`. All saved Test Run files are given a `.trf` file extension. In the file selection dialog box there is a file filter for `.trf` files. This allows easy access to saved Test Run files.

- Find the desired Test Run file.
- Select the Test Run file.
- Click `Open` to open the Test Run file.
- If you choose `Cancel`, the operation is cancelled and the file dialog box will be closed.

Once you have clicked `Open` and the file is a valid Test Run file, the Test Run will be opened and the Test Run added to the `Test Run's` tree node.

If you open a Test Run, when a Test Run with the same Test Run name is already open, a dialog box will be shown to ask if you really want to rename the nominated Test Run in order to open it. If the answer is `YES` a new dialog box will be displayed, to ask user for the new Test Run name, and the Test Run with the new Test Run name will be opened. If the answer is `NO` the operation will be cancelled.

Invalid Test Run File

If you choose to open a file that is not a valid Test Run file then the Load Execution problem dialog box will be displayed with the message: '`The System cannot find the file Specified`' and the file will NOT be opened.

Click on `OK`, and another dialog box will be displayed with the message '`Error: Test Run File [`*`file_name`*`] not valid`'. Click on OK again. Then you will need to select a new Test Run file.

3.1.2 Create a New Test Run

A new Test Run is one that is created from scratch. The User defines everything from the configuration setting to the scenario selection.

There are a number of ways to create a New Test Run:

- Use the icon in the toolbar 🗋 .
- Use the `New Test Run` item in the File menu.
- Right click on the `Test Run` tree node and choose `New Test Run`.
- Use the keyboard shortcut Alt-N.

Whichever method you choose to create a New Test Run you will then be prompted, by a dialog box, to specify the name of the Test Run. The name can be any string of characters.

Please note: Test Run names MUST be unique. This is to allow tracking of the Test Runs.

Once you click on OK, a check is made to see that the Test Run name does not already exist. If the Test Run name specified does not already exist then a new Test Run is created and added to the

Test Runs tree node. If the Test Run name does exist, then an error message is displayed, Click on OK to close the dialog box and choose another Test Run name.

3.2 Removing Test Run File(s).

To remove a `Test Run` from the tree you may follow any of the three methods below:

- Use the `Remove Test Run` menu item in the File menu.
- Right-click on the Test Run node in the tree and click the `Remove Test Run` menu item.
- Press the Delete key when the Test Run is highlighted.

Whichever way is chosen to remove the Test Run you will be prompted with a dialog box which ask you to Confirm removal of Test Run [*Name of Test Run*].

If the `Yes` button is clicked then all components of the Test Run will be removed from the tree. If `No` is chosen then the Test Run will not be removed and the process will be abandoned.

4 Configuration

The configuration Tree Node holds configuration variable assignments for each of TETware *professional*'s modes of operation. In addition, configuration files may hold configuration variable assignments for use by test cases and by build and clean tools.

Each Tree Node has its own specific configuration settings.

- Main configuration.
- Local System Configuration, including Embedded System configuration.
- Remote System Configuration (see Section 6).

4.1 Main Configuration Setting

To access the configuration screen you need to click on `Configuration` in the `Tree Node`, or press the up or down arrow keys once a tree node is highlighted.

Each Test Run must have one or more configuration files associated with it. These files contain configuration variable assignments, which are specified by the test suite author. When any test case is processed in a particular mode of operation, it reads variables from the configuration file for that mode.

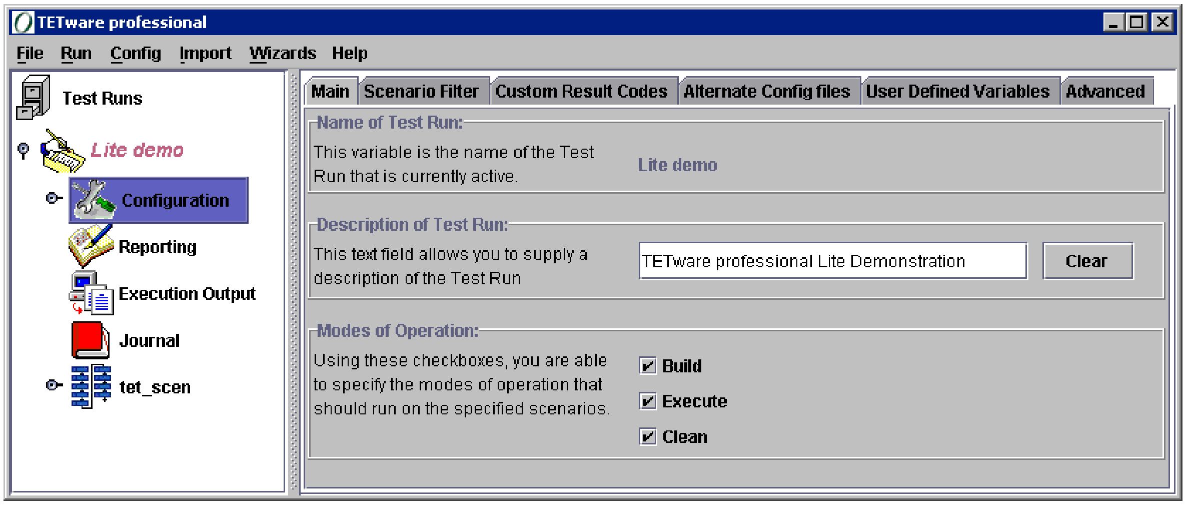
There are 6 tabs that can be accessed on the Configuration Screen:

- `Main` tab
- `Scenario Filter` tab
- `Custom Result Codes` tab
- `Alternate Config files` tab
- `User Defined Variables` tab
- `Advanced` tab

These tabs will now be described in detail.

4.1.1 Main Tab

The Main tab `Configuration` screen is displayed in Figure 3 below:



Name of Test Run:

This variable is the name of the Test Run that is currently active. Lite demo

Description of Test Run:

This text field allows you to supply a description of the Test Run TETware professional Lite Demonstration

Modes of Operation:

Using these checkboxes, you are able to specify the modes of operation that should run on the specified scenarios.

☑ Build
☑ Execute
☑ Clean

Figure 3: Main Configuration tab

This screen is broken down into a number of areas that can be chosen independently of the others.

The `Main` configuration tab holds the following values:

Name of Test Run As the name indicates, this field holds the name of the Test Run that is currently active. It is the name given to it by the user who first created the Test Run. It cannot be changed.

Description of Test Run This field holds a textual description of the Test Run, You are able to provide a 'one line' description that will give more information about a Test Run and it's purpose. You can edit the field by clicking in the box and editing or clear the contents using the `Clear` button.

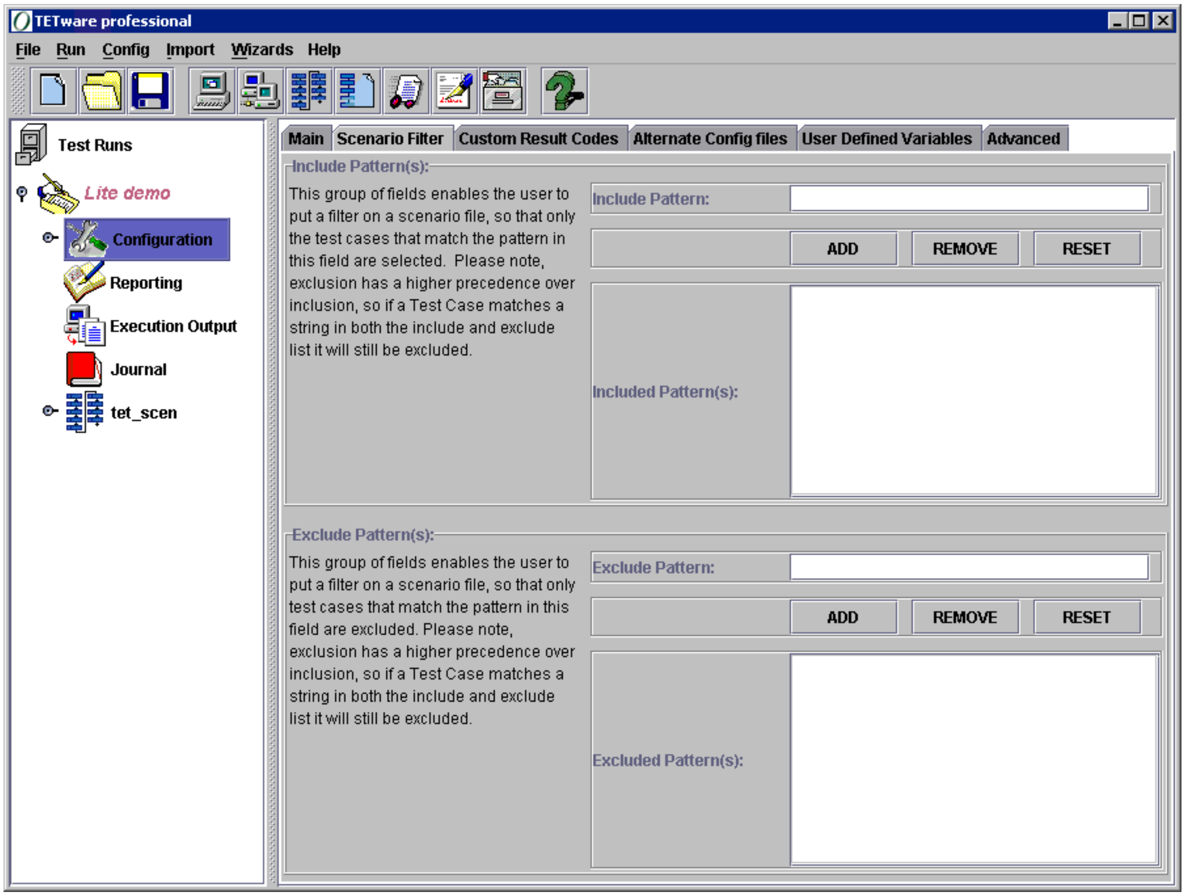
Modes of Operation This list of checkboxes allows you to choose the modes of operation that you wish TETware *professional* to use. Any combination of modes is valid although to successfully execute TETware *professional*, you MUST choose at least one mode.

4.1.2 Scenario Filter Tab

The Scenario filter tab allows you to specify Test Cases that you want to include and exclude from the Test Run.

Note: Exclusion has a higher precedence over inclusion, so if a Test Case matches a string in both the include and exclude lists it will be excluded.

You can specify any number of Test Cases to include or exclude from the Test Run. The Test Cases should be a string of characters that are either a regular expression or a sub-string of a Test Case name.



This group of fields enables the user to put a filter on a scenario file, so that only the test cases that match the pattern in this field are selected. Please note, exclusion has a higher precedence over inclusion, so if a Test Case matches a string in both the include and exclude list it will still be excluded.

This group of fields enables the user to put a filter on a scenario file, so that only test cases that match the pattern in this field are excluded. Please note, exclusion has a higher precedence over inclusion, so if a Test Case matches a string in both the include and exclude list it will still be excluded.

Figure 4: Scenario Filter

Include pattern(s): This group of fields enables the user to put a filter on a scenario file, so that only the test cases that match the pattern in this field are selected during the Test Run.

Exclude pattern(s): This group of fields enables the user to put a filter on a scenario file, so that only test cases that match the pattern in this field are excluded or skipped during Test Run execution.

The following keys are common to the two tabs

`ADD` button: You can add included or excluded Test Cases by typing the name (or regular expression) into the Include or Exclude Pattern field then press `ADD` to add the Test Case name to the Included or Exclude Pattern(s) list.

You can add multiple Test Cases to the Included or Excluded Pattern(s) list by pressing the space bar between the Test Case name (or regular expression) in the Include or Exclude Pattern field and pressing `ADD`.

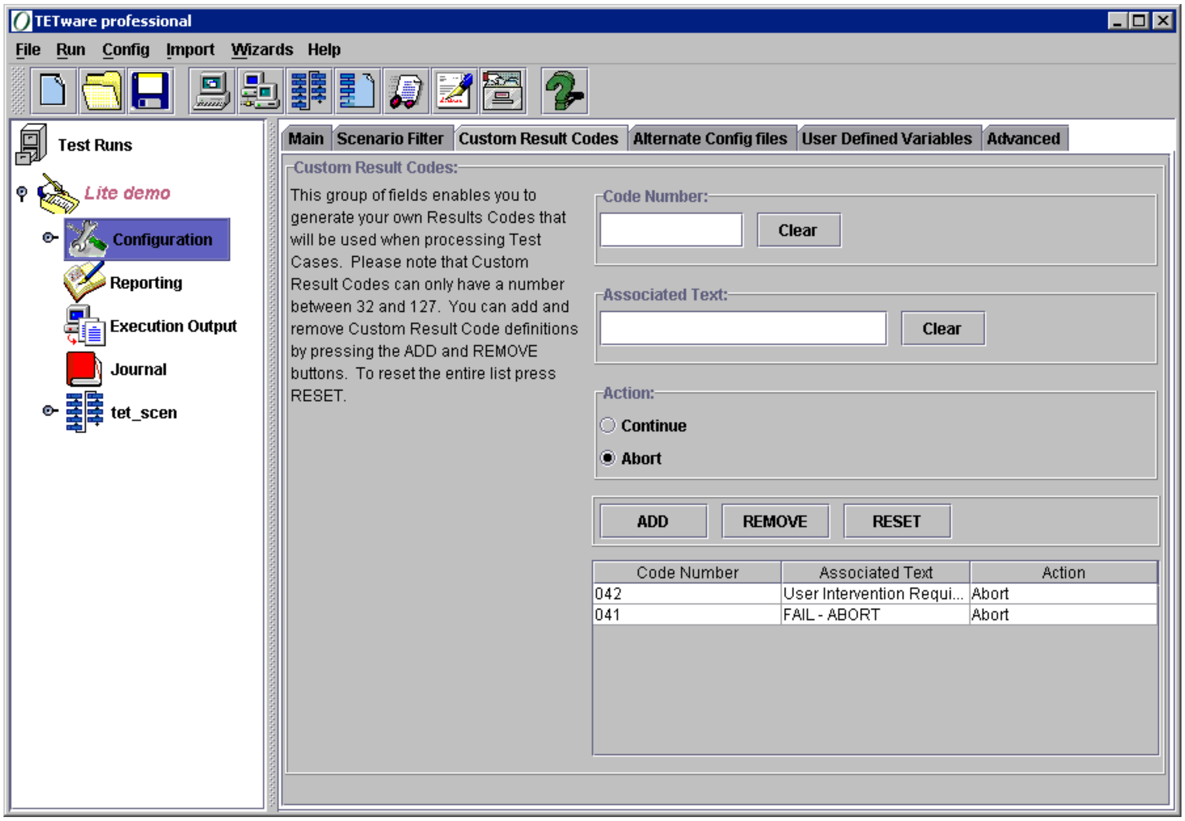
`REMOVE` button: To remove Test Cases from the Included or Excluded Pattern(s) list just select the Test Case you wish to remove and press `REMOVE`, the Test Case will then be removed.

To remove multiple Test Cases you can hold the `Shift` or `Ctrl` key and select rows in the list, if you hold '`Shift`' then you can select a contiguous list of Test Cases, if you hold `Ctrl` you can choose a dis-contiguous list of Test Cases. To remove the selected Test Cases press `REMOVE`.

`RESET` button: To remove all the Test Cases from the Included or Excluded Pattern(s) list just press the `RESET` button, all Test Cases will then be removed.

4.1.3 Custom Result Codes Tab

When TETware *professional* needs to interpret a test purpose result, it does so by referring to a table of results codes. Initially, this table contains a list of code values, which have standard meanings that may (optionally) be augmented by lists contained in files provided by the test developer.



This group of fields enables you to generate your own Results Codes that will be used when processing Test Cases. Please note that Custom Result Codes can only have a number between 32 and 127. You can add and remove Custom Result Code definitions by pressing the ADD and REMOVE buttons. To reset the entire list press RESET.

Action:

○ Continue

● Abort

Code Number	Associated Text	Action
042	User Intervention Requi...	Abort
041	FAIL - ABORT	Abort

Figure 5: Custom Result codes

The result code value is a positive number between 0 and 127. TETware *professional* reserves values between 0 and 31 for use. Of these Values 0 to 7 are currently used. These standard results codes, all have a Continue action indicator.

The Standard Result codes are the following:

0	=	PASS	4	=	UNSUPPORTED
1	=	FAIL	5	=	UNTESTED
2	=	UNRESOLVED	6	=	UNINITIATED
3	=	NOTINUSE	7	=	NORESULT

The Custom Results Codes tab allows you to specify Test Case Result Codes that are not part of the standard Result Codes that TETware *professional* use as it's defaults.

Custom Result Codes should have a value between 32 and 127. You can also give the Result Code some associated text. The value of the Result Code should be placed into the `Code Number` field and the Text relating to it in the `Associated Text` field. An action is also associated with the Result Code this is indicated by the ‘`Action`’ radio buttons; the options are `Continue` or `Abort`.

Adding Custom Result Codes:

To add a Result Code you must specify the `Code Number` and some `Associated Text`. If you do not specify an action then the default `Continue` will be used. To add the Result Code press `ADD`.

Added Result Codes are added to the table at the bottom of the tab. Once in the table, the values cannot be edited. However, they may be removed and replaced.

Removing Custom Result Codes:

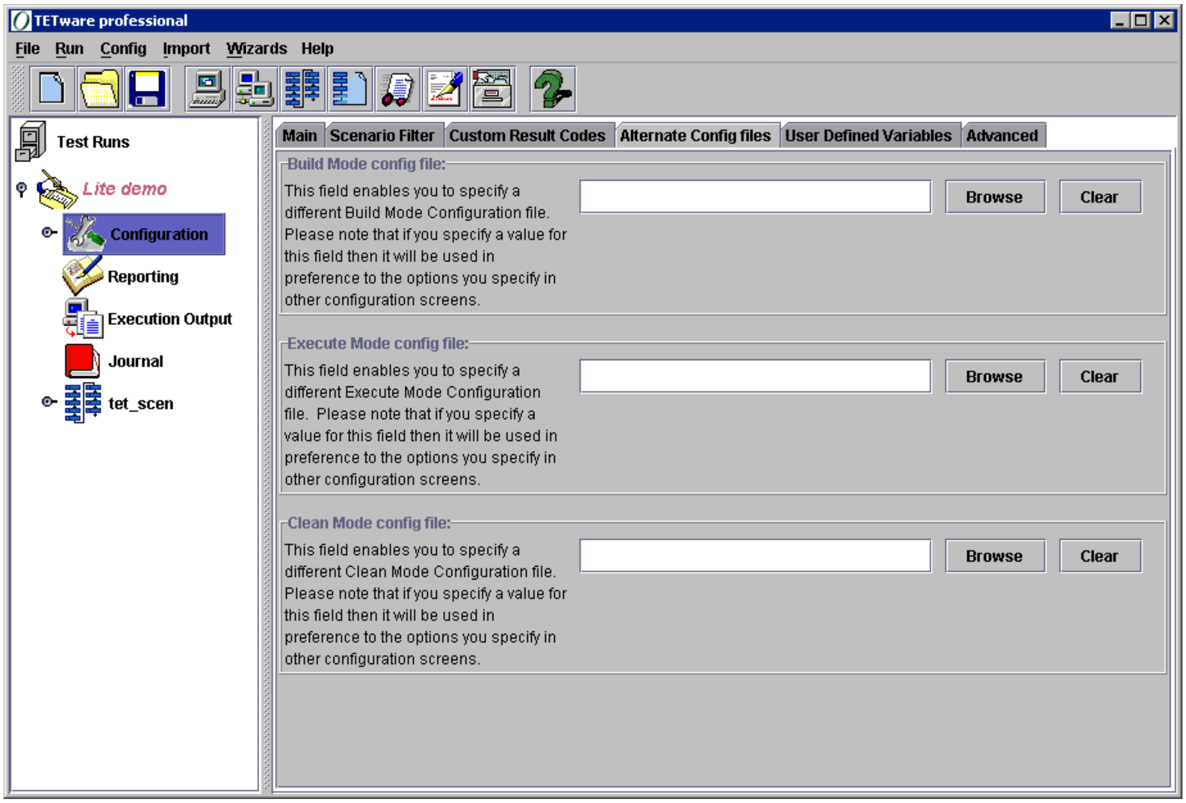
To remove Result Codes just click on the relevant row in the table and press `REMOVE`.

Reset

Pressing the `RESET` button will clear all the codes that were entered along with all the associated texts and actions.

4.1.4 Alternate Config Files Tab

You are able to specify alternative configuration files to use in place of the ones created by TETware *professional* when a Test Run is executed. These configuration files will be used and no configuration files will be generated by TETware *professional*. You can choose to use one or more alternative configuration files. If you do not specify a configuration file for a mode of operation then TETware *professional* will generate the file for you.



Build Mode config file:
This field enables you to specify a different Build Mode Configuration file. Please note that if you specify a value for this field then it will be used in preference to the options you specify in other configuration screens.

Execute Mode config file:
This field enables you to specify a different Execute Mode Configuration file. Please note that if you specify a value for this field then it will be used in preference to the options you specify in other configuration screens.

Clean Mode config file:
This field enables you to specify a different Clean Mode Configuration file. Please note that if you specify a value for this field then it will be used in preference to the options you specify in other configuration screens.

Figure 6: Alternate Config Files

The following alternative configuration files can be specified:

`Build` Specifying a valid file for the Build mode will force TETware *professional* to use this file rather than building a configuration file for the Build mode;

`Execute` Specifying a valid file for the Execute mode will force TETware *professional* to use this file rather than building an Execute mode configuration file;

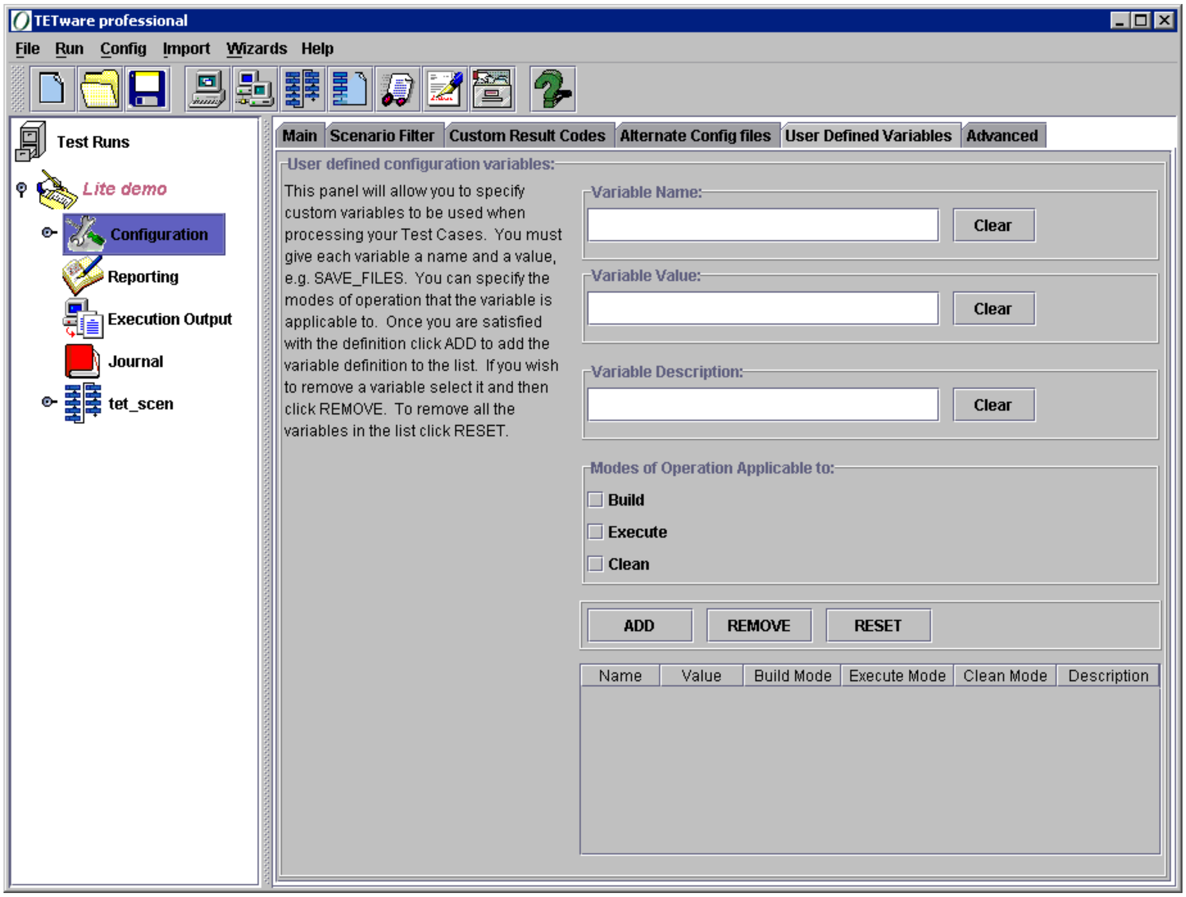
`Clean` Specifying a valid file for the Clean mode will force TETware *professional* to use this file rather than building a Clean mode configuration file.

4.1.5 User Defined Variables Tab

The User Defined Variables tab allows you to specify variables that will be used by your test code in carrying out the Test Run.

Each variable has a name and a value associated with it. In addition to this, a description of that value can be specified. Each variable is applicable to one or more Modes of Operation.

The User Defined Variables tab is shown below in Figure 7:



User defined configuration variables:

This panel will allow you to specify custom variables to be used when processing your Test Cases. You must give each variable a name and a value, e.g. SAVE_FILES. You can specify the modes of operation that the variable is applicable to. Once you are satisfied with the definition click ADD to add the variable definition to the list. If you wish to remove a variable select it and then click REMOVE. To remove all the variables in the list click RESET.

Variable Name:

Variable Value:

Variable Description:

Modes of Operation Applicable to:

□ Build

□ Execute

□ Clean

Name	Value	Build Mode	Execute Mode	Clean Mode	Description

Figure 7: User Defined Variables

4.1.5.1 Adding Variables

To add a new variable you need to specify:

Name	Required
Value	Required
Mode of Operation the variable is applicable to	Required
The description field	Optional

Once you are satisfied with the variable click `ADD` to add it to the list of variables. The variable will not be added to the table unless you have specified all the required values.

4.1.5.2 Removing Variables

To remove a variable select it from the table and click `REMOVE`, the variable will then be removed.

To remove multiple variables you can hold the ‘Shift’ or ‘Ctrl’ key and select rows in the table using your mouse or the arrow key, if you hold ‘Shift’ then you can select a contiguous list of variables but if you hold ‘Ctrl’, you can choose a non-continuous list of variables. To remove the selected Test Cases press `REMOVE` (please see Figure 7).

4.1.5.3 Removing all the Variables

To remove all the items from the table click `RESET`, this will remove all the defined variables from the table.

4.1.6 Advanced Tab

The Advanced tab is the place where all the values considered to be outside the normal scope of operation for TETware *professional* are placed. It may be that certain Users will use these features regularly however it is felt that overall these functions are ‘Advanced’.

The Advanced tab contains three groups of items namely:

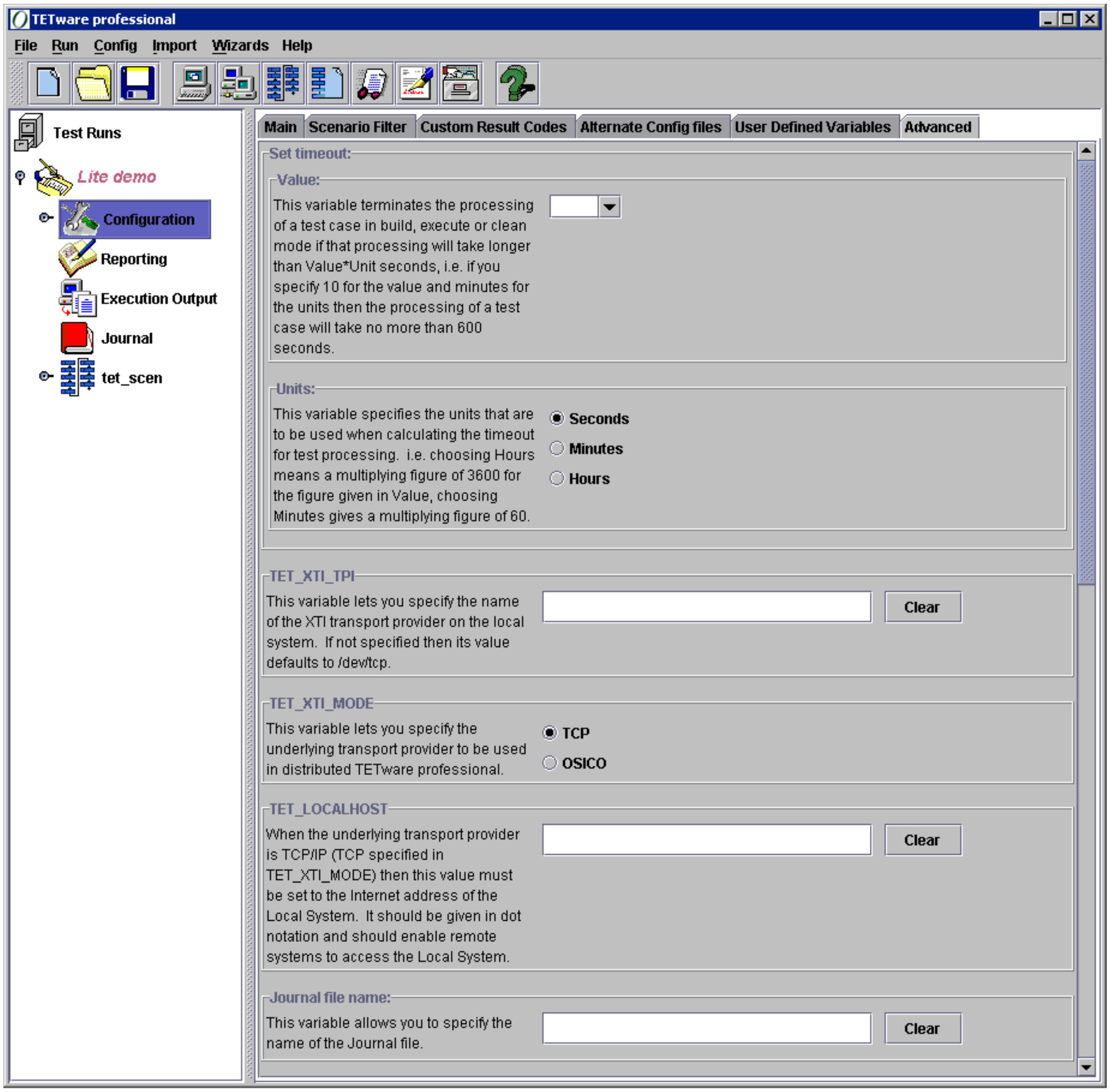
- Timeout
- Distributed network information
- User defined Journal file name

4.1.6.1 Timeout

You are able to specify a timeout that should be passed to TETware *professional* to indicate the amount of time that TETware *professional* should wait for a Test to complete.

You can specify a value for the Timeout. The Timeout fields are setup such that the you can specify your own Timeout value or you can select from a pre-defined value given in the drop down box, either 10, 20, 50, 100, 200. If you wish to specify Timeout, it can be typed directly into the first field of the drop down box. The value entered should be a maximum of 5 digits. If the value is invalid, or larger than 5 digits, an error message will be generated when you try to execute the test run, and the value will be ignored.

You are also able to specify a ‘`Units`’ value that will act as a multiplier for the Timeout value. The multipliers are Seconds, Minutes and Hours. Only one multiplier can be specified. Therefore, if you choose to have a timeout of 15 hours, you should type 15 into the value box and choose `Hours` as the `Units` value. When you execute the Test Run, TETware *professional* will determine the value of the Timeout in seconds (i.e. 15 x 60 x 60 = 54,000 Seconds).



Set timeout:

Value:

This variable terminates the processing of a test case in build, execute or clean mode if that processing will take longer than Value*Unit seconds, i.e. if you specify 10 for the value and minutes for the units then the processing of a test case will take no more than 600 seconds.

Units:

This variable specifies the units that are to be used when calculating the timeout for test processing. i.e. choosing Hours means a multiplying figure of 3600 for the figure given in Value, choosing Minutes gives a multiplying figure of 60.

● Seconds
○ Minutes
○ Hours

TET_XTI_TPI

This variable lets you specify the name of the XTI transport provider on the local system. If not specified then its value defaults to /dev/tcp.

TET_XTI_MODE

This variable lets you specify the underlying transport provider to be used in distributed TETware professional.

● TCP
○ OSICO

TET_LOCALHOST

When the underlying transport provider is TCP/IP (TCP specified in TET_XTI_MODE) then this value must be set to the Internet address of the Local System. It should be given in dot notation and should enable remote systems to access the Local System.

Journal file name:

This variable allows you to specify the name of the Journal file.

Figure 8: Advanced tab

4.1.6.2 Distributed Network Information

If distributed TETware *professional* is used then it may be that information regarding the Local Systems network address and underlying network protocol is required.

`TET_XTI_TPI` This variable lets you specify the name of the XTI transport provider on the Local System. If this is not specified, the value defaults to `/dev/tcp`.

`TET_XTI_MODE` This variable lets you specify the underlying transport provider to be used in TETware *professional*. There are two choices, either `TCP` or `OSICO.`

`TET_LOCALHOST` When the underlying transport provider is TCP/IP (TCP specified in `TET_XTI_MODE`) then this value must be sent to the Internet address of the Local System. It should be given in dot notation and should enable the Remote Systems to access the Local System.

4.1.6.3 Journal file name

This field allows the User to specify the name of the Journal file that TETware *professional* should use for Journal file entries. Please note that you must specify the path, the file name and its extension.

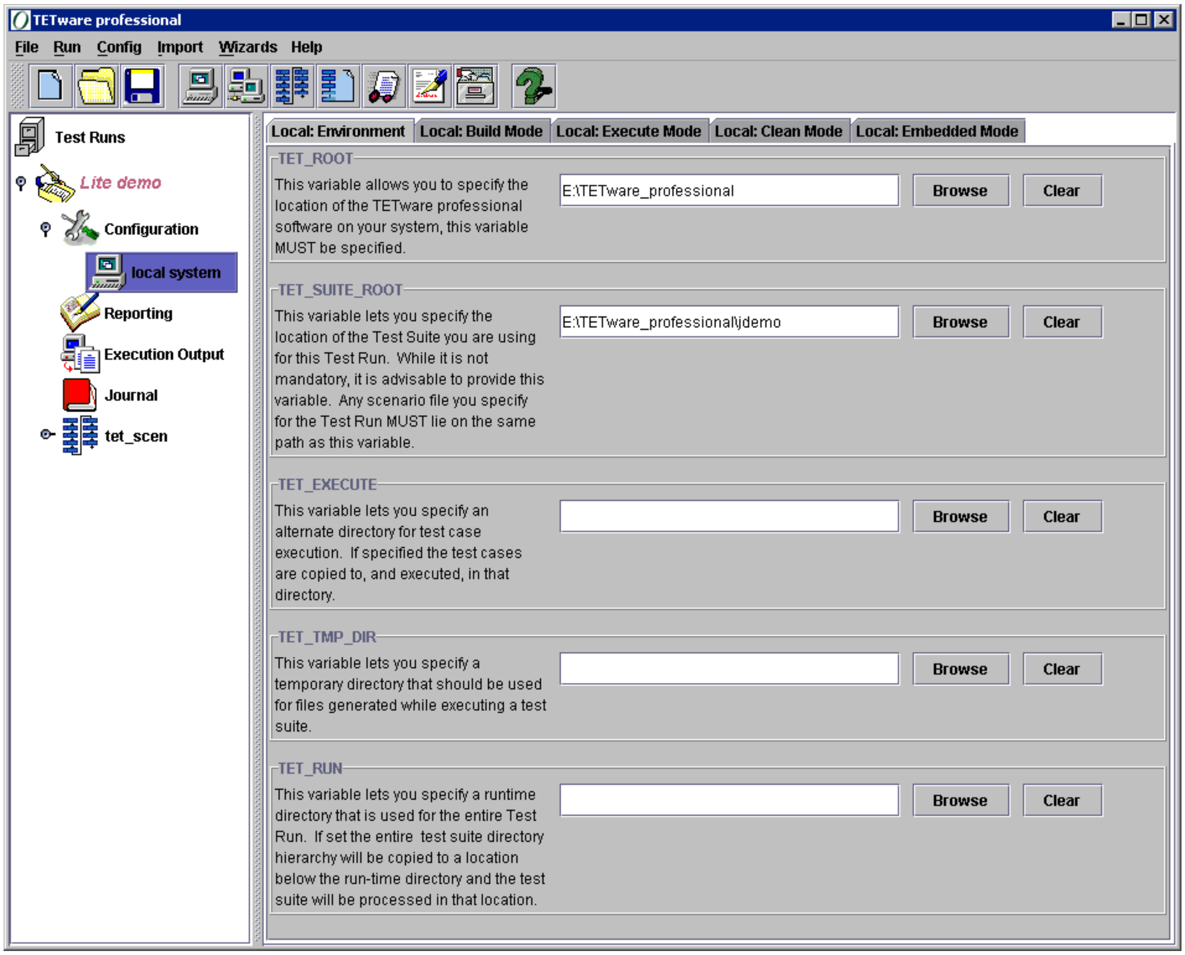
4.2 Local System Configuration

Configuration variables may be specified both on the Local System, and also for Embedded Realtime Systems if TETware *professional* is used to control the execution of Realtime System testing. The configuration setting on the Local System is divided into five tabs:

- Local: Environment
- Local: Build Mode
- Local: Execute Mode
- Local: Clean Mode
- Local: Embedded Mode

4.2.1 Local: Environment Tab[1]

This, for the Local System, is the place to setup all the environment variables that TETware *professional* will and may need to execute correctly.



TET_ROOT

This variable allows you to specify the location of the TETware professional software on your system, this variable MUST be specified.

E:\TETware_professional

TET_SUITE_ROOT

This variable lets you specify the location of the Test Suite you are using for this Test Run. While it is not mandatory, it is advisable to provide this variable. Any scenario file you specify for the Test Run MUST lie on the same path as this variable.

E:\TETware_professional\jdemo

TET_EXECUTE

This variable lets you specify an alternate directory for test case execution. If specified the test cases are copied to, and executed, in that directory.

TET_TMP_DIR

This variable lets you specify a temporary directory that should be used for files generated while executing a test suite.

TET_RUN

This variable lets you specify a runtime directory that is used for the entire Test Run. If set the entire test suite directory hierarchy will be copied to a location below the run-time directory and the test suite will be processed in that location.

Figure 9: Local: Environment tab

[1] Please note: all variables specified in this tab should be directories, when the Test Run is executed a check is made to ensure that all variables specified in this tab are directories, an error will be displayed if one or more variables are NOT directories and the Test Run execution halted.

All the variables specified in this tab will be passed to TETware *professional* as Environment variables. That is, they will exist as part of TETware *professional*'s run-time environment only for the duration that TETware *professional* is running. Any other Test Runs that are executing can and will have their own environment specified.

The following variables can be specified in the Local: Environment tab:

TET_ROOT This variable allows the user to specify to TETware *professional* the location of the software, library files and certain configuration files on the users system. This variable MUST always be specified for the Local System.

TET_SUITE_ROOT This variable allows the user to specify to TETware *professional* the location of the Test Suite that is to be used for the Test Run. In general, TET_SUITE_ROOT would sit below TET_ROOT in the directory structure but this is not always true. The Scenario file added to the Test Run must lie somewhere in the directory structure below TET_SUITE_ROOT. This variable is optional but necessary if the Test Suite does not lie below TET_ROOT in the file system directory structure.

TET_EXECUTE This variable lets you specify an alternate directory for test case execution. If specified the test cases are copied to, and executed in, that directory.

TET_TMP_DIR This variable lets you specify a temporary directory that should be used for files generated while executing a test suite.

TET_RUN If specified this variable indicates to TETware *professional* that it should copy the entire Test Suite to the directory specified and process the Test Suite in that directory.

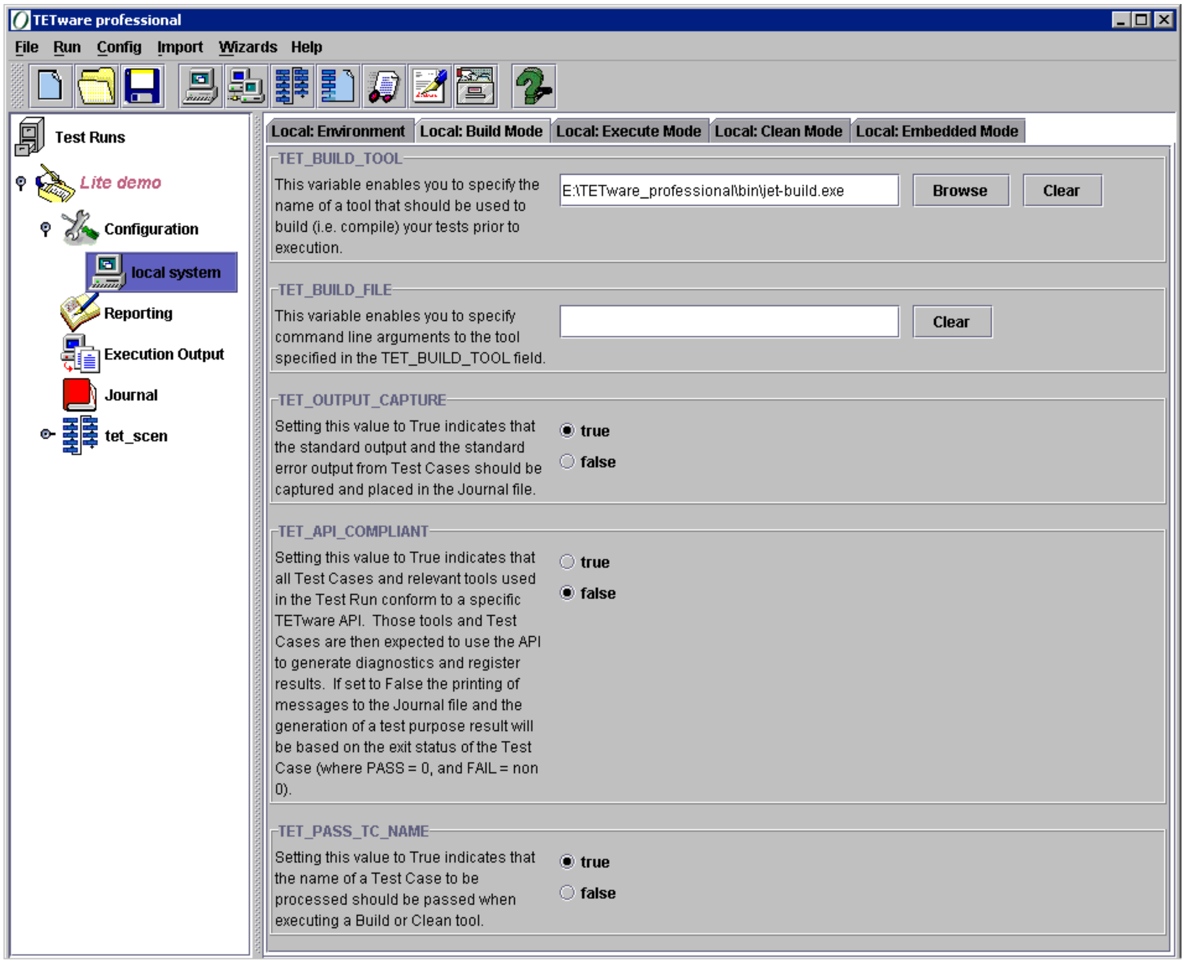
4.2.2 Local: Build Mode

This tab allows you to specify variables that will be used by TETware *professional* in Build mode.

The following variables can be specified in Local: Build Mode:

TET_BUILD_TOOL This variable enables you to specify a tool that TETware *professional* should use to build (i.e. compile) test cases, for example make.

TET_BUILD_FILE This variable allows you to specify additional command line arguments that can be passed to the build tool specified by TET_BUILD_TOOL. Please note: no validation is performed by TETware *professional* for the relationship between TET_BUILD_TOOL and TET_BUILD_FILE; it is the Users responsibility to ensure that they interact correctly.



TET_BUILD_TOOL

This variable enables you to specify the name of a tool that should be used to build (i.e. compile) your tests prior to execution.

E:\TETware_professional\bin\jet-build.exe

TET_BUILD_FILE

This variable enables you to specify command line arguments to the tool specified in the TET_BUILD_TOOL field.

TET_OUTPUT_CAPTURE

Setting this value to True indicates that the standard output and the standard error output from Test Cases should be captured and placed in the Journal file.

● true
○ false

TET_API_COMPLIANT

Setting this value to True indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to False the printing of messages to the Journal file and the generation of a test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

○ true
● false

TET_PASS_TC_NAME

Setting this value to True indicates that the name of a Test Case to be processed should be passed when executing a Build or Clean tool.

● true
○ false

Figure 10: Local: Build Mode

`TET_OUTPUT_CAPTURE` These radio buttons indicate to TETware *professional* that the standard output and error streams from the Build tool should be captured and placed in the Journal file. The default value of this variable is `True`. Setting this variable to `False` tells TETware *professional* not to record test case output in the journal file.

`TET_API_COMPLIANT` These radio buttons allow you to specify to TETware *professional* whether the Build tool and Test Cases are to be built to conform to TETware *professional* API, i.e. they use TETware *professional* API to output information to the Journal file. The default value of this variable is `False`. Setting the value to `False` indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to `True` the printing of messages to the journal file and the generation of the test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

TET_PASS_TC_NAME These radio buttons allow you to specify to TETware *professional* whether the name of the current Test Case should be passed to the Build tool specified. The default value of this variable is `False`. Setting this value to `True` indicates that the name of the Test Case to be processed should be passed when executing a Build or Clean tool.

4.2.3 Local: Execute Mode

This tab allows you to specify variables that will be used by TETware *professional* in Execute mode. Please see Figure 11.

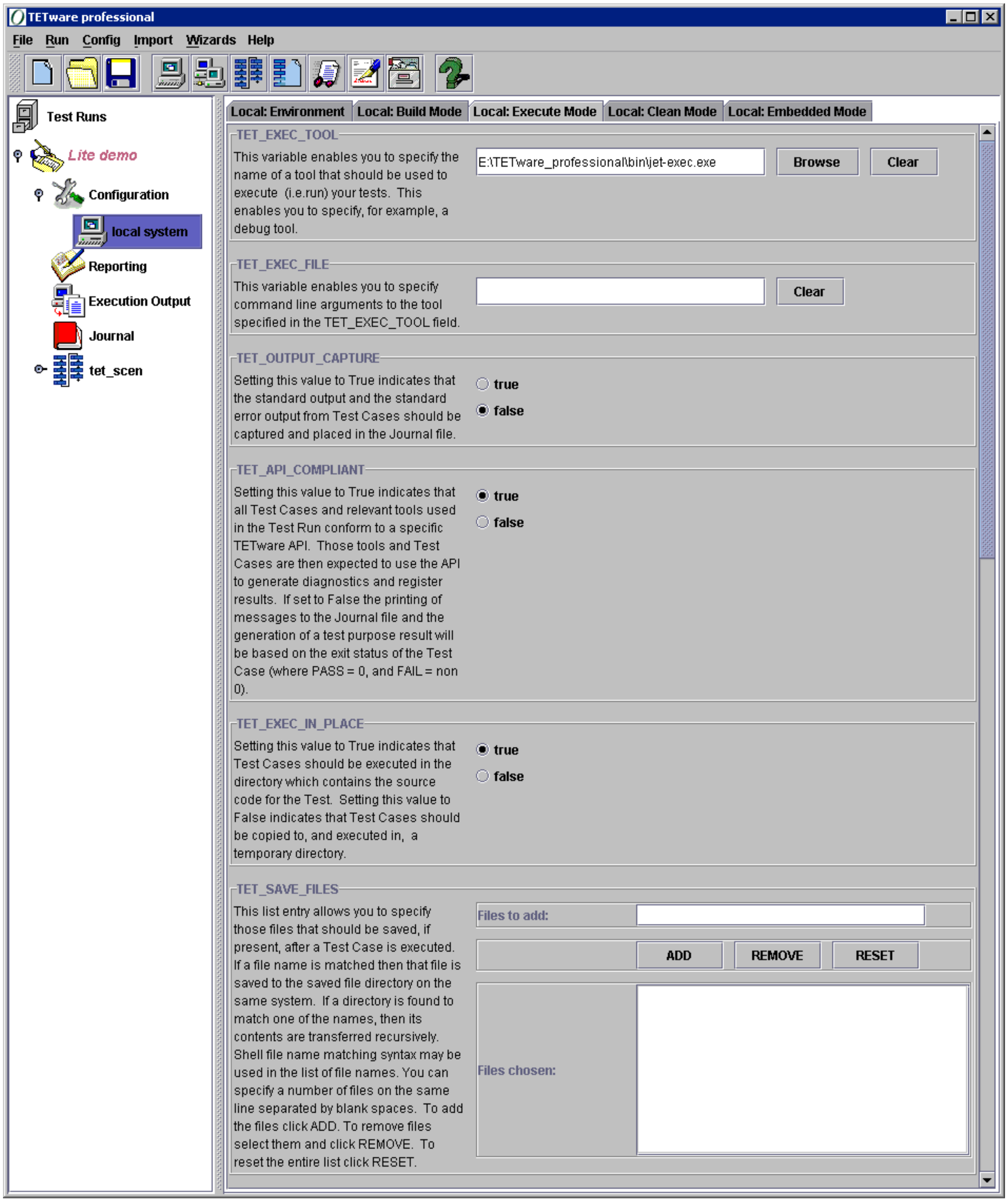
TET_EXEC_TOOL This variable enables you to specify a tool that TETware *professional* should use to execute test cases.

TET_EXEC_FILE This variable allows you to specify additional command line arguments that can be passed to the execute tool specified by `TET_EXEC_TOOL`. Please note: no validation is performed by TETware *professional* for the relationship between `TET_EXEC_TOOL` and `TET_EXEC_FILE`; it is the Users responsibility to ensure that they interact correctly.

TET_OUTPUT_CAPTURE This radio button variable indicates to TETware *professional* whether the standard output and error streams from the Execute tool should be captured and placed in the Journal file. The default value of this variable is `True`. Setting this value to `False` indicates that the standard output and the standard error output from Test Cases should not be captured and placed in the journal file.

TET_API_COMPLIANT This radio button variable allows you to specify to TETware *professional* whether the Execute tool and Test Cases to be built conform to TETware *professional* API, i.e. they use the TETware *professional* API to output information to the Journal file. The default value of this variable is `False`. Setting this value to `True` indicated that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to `False` the printing of the journal file and the generation of a Test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

TET_EXEC_IN_PLACE These radio buttons indicate to TETware *professional* that it should execute Test Cases in the directory that contains the source code for the Tests. Setting this value to `True` indicates that Test cases should be executed in the directory, which contains the source code for the Test. Setting the value to `False` indicates that the Test Case should be copied to, and executed in, a temporary directory.



TET_EXEC_TOOL

This variable enables you to specify the name of a tool that should be used to execute (i.e.run) your tests. This enables you to specify, for example, a debug tool.

E:\TETware_professional\bin\jet-exec.exe

TET_EXEC_FILE

This variable enables you to specify command line arguments to the tool specified in the TET_EXEC_TOOL field.

TET_OUTPUT_CAPTURE

Setting this value to True indicates that the standard output and the standard error output from Test Cases should be captured and placed in the Journal file.

○ true
● false

TET_API_COMPLIANT

Setting this value to True indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to False the printing of messages to the Journal file and the generation of a test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

● true
○ false

TET_EXEC_IN_PLACE

Setting this value to True indicates that Test Cases should be executed in the directory which contains the source code for the Test. Setting this value to False indicates that Test Cases should be copied to, and executed in, a temporary directory.

● true
○ false

TET_SAVE_FILES

This list entry allows you to specify those files that should be saved, if present, after a Test Case is executed. If a file name is matched then that file is saved to the saved file directory on the same system. If a directory is found to match one of the names, then its contents are transferred recursively. Shell file name matching syntax may be used in the list of file names. You can specify a number of files on the same line separated by blank spaces. To add the files click ADD. To remove files select them and click REMOVE. To reset the entire list click RESET.

Files to add:

Files chosen:

Figure 11: Local: Execute Mode

The following variables can be specified in the Local: Execute Mode tab:

TET_SAVE_FILES You can specify a number of files that TETware *professional* should save after Test execution (if it is able to find those files). You can specify directories or normal shell matching syntax (i.e. *) as part of the list. To add the files enter them in the 'Files to add' text field and then press ADD.

To remove files, select the rows in the list and press REMOVE; you can select a contiguous list of files by pressing and holding 'Shift' whilst clicking the files. You can select a dis-contiguous list of files by pressing and holding 'Ctrl' whilst clicking the files.

The RESET button will remove the entire chosen file and leave the dialog box in its original position thereby allowing the user to start over again.

4.2.4 Local: Clean Mode

This tab allows you to specify variables that will be used by TETware *professional* in Clean mode. (Please see Figure 12).

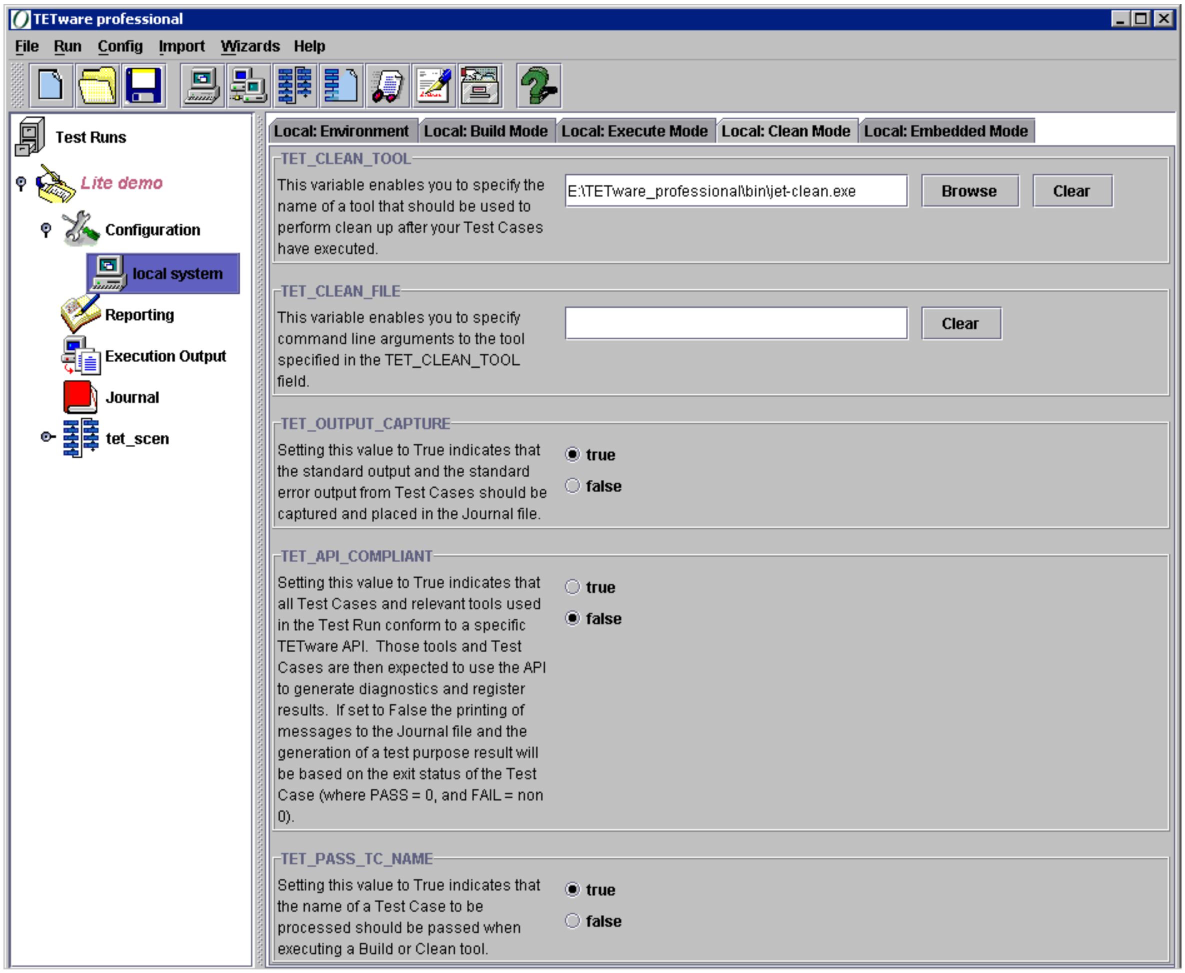
The following variables can be specified in the Local: Clean Mode tab:

TET_CLEAN_TOOL This variable enables you to specify a tool that TETware *professional* should use to clean test cases.

TET_CLEAN_FILE This variable allows you to specify additional command line arguments that can be passed to the clean tool specified by TET_CLEAN_TOOL. Please note: no validation could be performed by TETware *professional* for the relationship between TET_CLEAN_TOOL and TET_CLEAN_FILE; it is the Users responsibility to ensure that they interact correctly.

TET_OUTPUT_CAPTURE These radio buttons indicate to TETware *professional* whether the standard output and error streams from the Clean tool should be captured and placed in the Journal file. The default value of this variable is True. Setting this value to False indicates that the standard output and the standard error output from Test Cases should not be captured and placed in the journal file.

TET_API_COMPLIANT These radio buttons allows you to specify to TETware *professional* whether the Clean tool and Test Cases to be built conform to TETware *professional* API, i.e. they use TETware *professional* API to output information to the Journal file. The default value of this variable is `False`. Setting the value to `True` indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to `False` the printing of messages to the journal file and the generation of the test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).



TET_CLEAN_TOOL

This variable enables you to specify the name of a tool that should be used to perform clean up after your Test Cases have executed.

E:\TETware_professional\bin\jet-clean.exe

TET_CLEAN_FILE

This variable enables you to specify command line arguments to the tool specified in the TET_CLEAN_TOOL field.

TET_OUTPUT_CAPTURE

Setting this value to True indicates that the standard output and the standard error output from Test Cases should be captured and placed in the Journal file.

● true
○ false

TET_API_COMPLIANT

Setting this value to True indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to False the printing of messages to the Journal file and the generation of a test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

○ true
● false

TET_PASS_TC_NAME

Setting this value to True indicates that the name of a Test Case to be processed should be passed when executing a Build or Clean tool.

● true
○ false

Figure 12: Local: Clean Mode.

TET_PASS_TC_NAME These radio buttons allow you to specify to TETware *professional* whether the name of the current Test Case should be passed to the Clean tool specified. The default value of this variable is FALSE. Setting this value to TRUE indicates that the name of a Test Case to be processed should be passed when executing a Build or Clean tool.

4.2.5 Local: Embedded Mode

This tab allows users to specify variables that TETware *professional* uses in execute mode to determine how to process a test case on an Embedded Realtime System.

Please refer to the TETware *professional* Realtime and Embedded Systems Extension; Installation, User and Programmer Guide for details.

4.3 Importing Configuration files

For existing TETware and TET Users an import function has been provided for importing configuration values. Importing configuration is performed on a per Test Run basis. The following types of configuration file are supported for import:

- Build mode
- Execute mode
- Clean mode
- Distributed config

For each configuration file imported the file is parsed and values that TETware *professional* understands are placed into the configuration for the given Test Run. Values that TETware *professional* does not understand are added to the User defined variables for a Test Run. All lines in the configuration file that start with # are ignored as comment lines. Comment lines are not imported since it is impossible to define rules that would capture the many different ways of commenting that are possible.

4.3.1 User Defined Variables

Imported User defined values will be added to the table in the User Defined variables tab, for more information on User Defined Variables please see Section 4.1.5. The mode of operation that is given to the imported User defined variable is determined by the type of configuration file that is imported. For instance, if a Build mode configuration file is imported then all User Defined variables in that file will be given a mode of Build in the User defined variables table.

4.3.2 Distributed config files

It is also possible to import a distributed configuration file (usually called `tetdist.cfg`). When importing a distributed configuration file you MUST have all the remote systems that have values in the distributed configuration file already added to the Test Run. i.e. if the distributed configuration file contains a value `TET_REM001_TET_ROOT` then a remote system with a system number of 001 must already be present in the Test Run for that value to be imported correctly.

4.3.3 How to import a configuration file

To import a configuration file you can either:

- Use the Import menu on the menu bar, from there you can choose either Build, Execute, Clean or Distributed configuration files to import.

- Right-click on the `Configuration` node in the Test Run. A pop-up menu containing items for Build, Execute, Clean and Distributed configuration files are shown.

Regardless of the method of import, a file selection dialog is then displayed from which the User can browse the file system and choose a configuration file. A file filter of `*.cfg` is also given to filter for configuration files that are given the default extension of cfg.

Please note: the parser for the configuration files will split any valid lines (lines not blank and not beginning with #) on "=", so any malformed lines will be added without verification. It is the Users responsibility to ensure that the configuration file is correct and contains values that are valid and have meaning to TETware *professional* and their Test Cases.

5 Scenario File(s)

To execute a Test Run, a scenario file MUST be included.

5.1 What is a Scenario File?

For a `Test Run` to be useful, one or more scenarios must be selected from a Scenario file that has been added into the `Test Run`. Scenario files are used in manipulating the `Test Run`.

By default there is a scenario called `all`. This `all` scenario has a special meaning to TETware *professional*, if it is defined in the scenario file then it will be mutually exclusive with the other scenarios in the scenario file. This is to prevent locking problems and potential problems with scenario looping. If the `all` scenario is chosen then all other scenarios will be set to not selected and the `all` scenario - only - will be selected. If another Scenario is selected while the `all` scenario is selected, the `all` Scenario will be set to not selected and the chosen scenario will be selected.

It is possible to select all of the scenarios in a Scenario file by right clicking on the Scenario node in the tree and clicking `Include all scenarios`, this will set all of the scenarios in the Scenario file as selected.

5.2 Adding a Scenario File

To add a scenario file the User can:

- Use the icon on the tool bar .
- Use the `Add Scenario File` menu item in the `Config` menu.
- Right-click on the Test Run tree node and click on the `Add Scenario File` menu item.[2]

[2] Please note that these menu items will only be available if there is NOT a scenario file already added to the Test Run.

Whichever method is chosen, a file selection dialog box will be displayed for you to choose the Scenario File needed.

There are two file filters provided to allow the User to find and select scenario files quickly, they are:

`scen.*` This will filter for all files STARTING with `scen.`
`*.scn` This will filter for all files ENDING with `.scn`.

Once a scenario file has been selected by clicking `Open`, a check will be made to ensure that the chosen file is not a directory and that it exists. If those criteria are NOT met then a dialog box will be displayed with a message ‘Scenario file is not valid – Error [*File name*] is not a valid file’.

If a file is chosen and it is valid, (i.e. it exists and is a file not a Directory) then it will be parsed by the Scenario file parser. The name of the Scenario file will be added to the Scenario file node and all the scenarios found added to that Node as children[3]. When a scenario file is added, the top scenario (usually “all”) is selected (for details of the “all” scenario please see Section 5.1). Note that only one scenario file may be included in a Test Run.

5.3 Selecting Test Cases from a Scenario File

To select Test Cases from within a Scenario file, you will need to

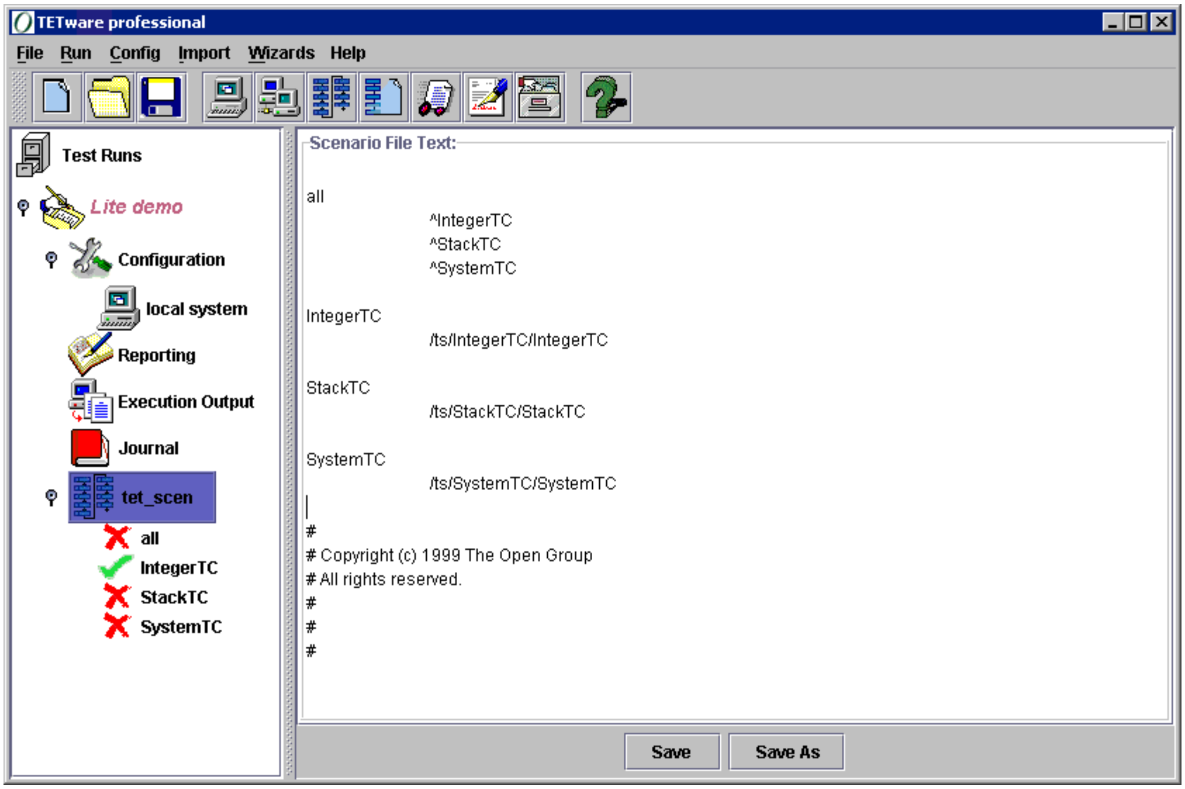
Expand the Scenario Node

- Place your Cursor on the Test Case to select.
- Click on the test case.
- The selected case will then change from a ✗ to a ✓.

[3] Please note: once a Scenario file has been added to the Test Run, the menu items for adding a Scenario file will be disabled until that Scenario file is removed.

5.4 Editing a Scenario File

To Edit a Scenario file, you will need to expand the Scenario Node.



```
all
                ^IntegerTC
                ^StackTC
                ^SystemTC

IntegerTC
                /ts/IntegerTC/IntegerTC

StackTC
                /ts/StackTC/StackTC

SystemTC
                /ts/SystemTC/SystemTC

#
# Copyright (c) 1999 The Open Group
# All rights reserved.
#
#
#
```

Figure 13: Editing Scenario File

- Click on a selected Scenario or on any of the expanded Test Cases
- Place your cursor on the Test Case field
- Start editing by typing any changes you wish to make.
- Then click on SAVE to save using the existing scenario file name or click on the SAVE AS button to save using another name.

If SAVE is selected a message box confirms the name of the saved Scenario file. If SAVE AS is selected a file chooser box appears giving the user an opportunity to save the edited Scenario file using another name.

5.5 De-Selecting Test Cases from the Scenario File

To De-select Test Cases from within a Scenario file, you will need to:

- Expand the Scenario Node
- Place your Cursor on the `Test Case` to select
- Click on the `Test Case`
- The selected case should then change from a ✓ to a ✗ to show that it has been deselected.

5.6 Creating a Scenario File

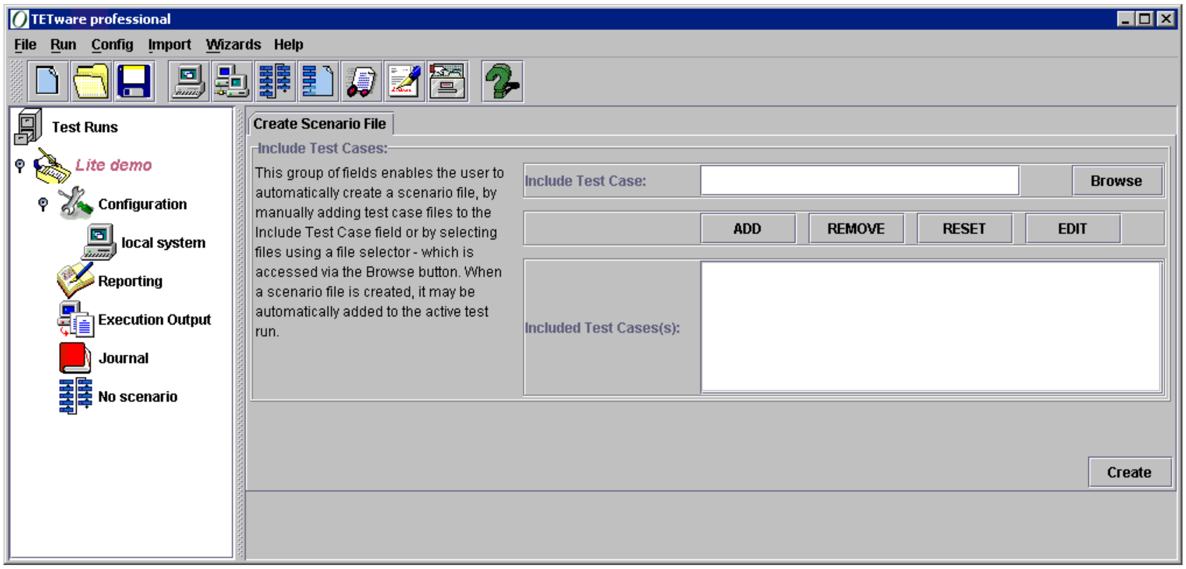
A scenario file can be created automatically by using a file parser. The generated scenario file is in a simple format, without directives. It can be edited and saved, including saved as a different file name if the user wishes.

To Create a Scenario File you can:

- Use the icon on the menu bar.
- Use the `Create Scenario File` menu item in the `Config` menu.
- Right-click on the Test Run tree node at the `No Scenario` point and click on the `Create Scenario File` menu item.

(Please note that these items will only be available if there is NOT a scenario file already added to the Test Run)

Whichever method is chosen, the create Scenario file tab is displayed.



This group of fields enables the user to automatically create a scenario file, by manually adding test case files to the Include Test Case field or by selecting files using a file selector - which is accessed via the Browse button. When a scenario file is created, it may be automatically added to the active test run.

Figure 14: Create Scenario file Tab

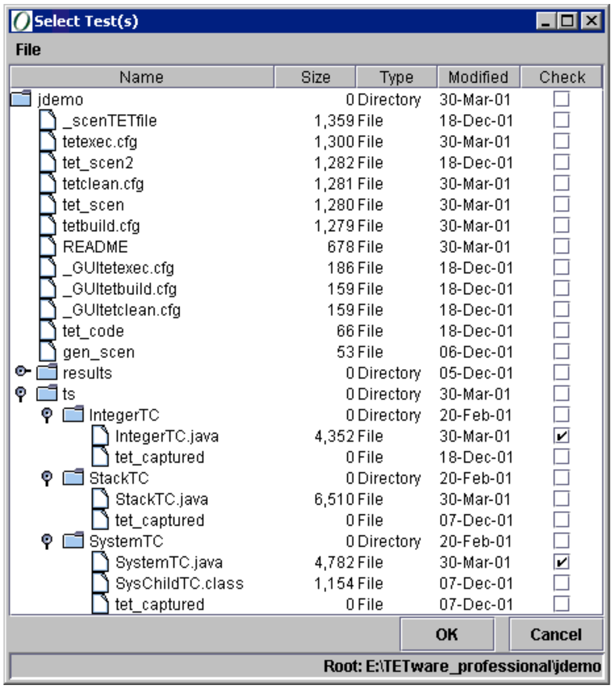
5.6.1 Including Test Cases in the Scenario File Creation (ADD)

There are two main ways of creating a Scenario file on this screen, either

- Type in the path and name of the Test Case(s) or
- Browse and select Test Case(s).

(Please note that, you need to have specified where TET_ROOT is, under Local System: Environment TET_ROOT settings, see Section 4.2.1).

- If you choose to type into the field `Include Test Cases`, you will need to click on the ADD button and the Test Case will be included in the scenario file being created.
- If you choose to browse, then the Select Test(s) dialog box opens, which will allow you to choose test cases by putting a check mark in the appropriate check box.



Name	Size	Type	Modified	Check
jdemo	0	Directory	30-Mar-01	□
_scenTETfile	1,359	File	18-Dec-01	□
tetexec.cfg	1,300	File	30-Mar-01	□
tet_scen2	1,282	File	18-Dec-01	□
tetclean.cfg	1,281	File	30-Mar-01	□
tet_scen	1,280	File	30-Mar-01	□
tetbuild.cfg	1,279	File	30-Mar-01	□
README	678	File	30-Mar-01	□
_GUItetexec.cfg	186	File	18-Dec-01	□
_GUItetbuild.cfg	159	File	18-Dec-01	□
_GUItetclean.cfg	159	File	18-Dec-01	□
tet_code	66	File	18-Dec-01	□
gen_scen	53	File	06-Dec-01	□
results	0	Directory	05-Dec-01	□
ts	0	Directory	30-Mar-01	□
IntegerTC	0	Directory	20-Feb-01	□
IntegerTC.java	4,352	File	30-Mar-01	☑
tet_captured	0	File	18-Dec-01	□
StackTC	0	Directory	20-Feb-01	□
StackTC.java	6,510	File	30-Mar-01	□
tet_captured	0	File	07-Dec-01	□
SystemTC	0	Directory	20-Feb-01	□
SystemTC.java	4,782	File	30-Mar-01	☑
SysChildTC.class	1,154	File	07-Dec-01	□
tet_captured	0	File	07-Dec-01	□

Root: E:\TETware_professional\jdemo

Figure 15: Scenario File Chooser

- You should then click on `OK` and the file(s) will be transferred to the Included Test Case(s) field. Alternatively, if you click on `Cancel`, the Select Test(s) box is closed and you are returned to the Create Scenario File tab.

5.6.2 Removing Test Cases from Scenario File Creation (`REMOVE`)

To remove any of the previously added Test cases in the Scenario file, you will need to select them from the Included Test Case(s) field and click `REMOVE`, the files will then be removed.

To remove multiple files you can hold the '`Shift`' or '`Ctrl`' key and select rows in the table using your mouse or the arrow key, if you hold '`Shift`' then you can select a contiguous list of variables but if you hold '`Ctrl`', you can choose a non-contiguous list of files. To remove the selected files, press `REMOVE`.

5.6.3 Resetting the Whole File Selection Process (`RESET`)

If you want to reset the whole file selection, all you need to do is to press on the `RESET` button and all the included test case(s) will disappear leaving the field empty.

5.6.4 Editing the Selected Variable (`EDIT`)

If you wish to change some of the parameters in your selection, you should highlight the file to be edited, and click on the `EDIT` button. This will cause the file to appear in the field `Include Test Case` field, it can then be edited in this field.

After editing, you will again need to press on the `ADD` button to include the edited file in the scenario that is being created.

5.6.5 Saving the Scenario File (`CREATE`)

After you have browsed, edited, and added the selected test cases into the scenario file to be created. You need to click on the `Create` button to generate and save the Scenario file.

When the `Create` button is pressed, a Save As dialog box will appear to provide the option to save the scenario file into any drive and directory. You should save it in either your `TET_ROOT` or `TET_SUITE_ROOT` directory.

After making your selection and naming the scenario file created, a confirmation dialog box `Save Scenario File` will be presented. Click on `OK` to close it.

A dialog box is then presented providing the option of adding the new Scenario file to the Test Run tree. If you select `Yes`, the Scenario file is added to the tree and the generated Scenario, `all` is selected. Selecting `No` closes the dialog box and returns you to the `Create Scenario` File tab.

Once added the Scenario file may be edited in the Scenario File Text screen (see Section 5.4) and can be saved using the same name, or saved as an alternative file name.

5.7 Removing a Scenario File

There are three ways to remove a Scenario file:

- Use the `Remove Scenario File` menu item in the `Config` menu.
- Right-click on the scenario node in the tree and click the `Remove Scenario File` menu item.
- Press the Delete key when the Scenario file is highlighted.

Whichever method is used, a dialog box will be displayed for you to confirm removal of the Scenario file.

If the `Yes` button is clicked then the Scenario file will be removed from the Test Run. All the child scenario nodes will also be removed. The label on the Scenario file node in the Test Run tree will be set to `No Scenario`. If `No` is clicked then the Scenario file will not be removed from the Test Run. The Scenario Node settings will remain as they were.

Once the Scenario file has been removed, the menu items relating to the addition of a Scenario file will be re-activated, and the menu items relating to remove a Scenario file will be de-activated.

6 Remote Systems

Configuration variables may be specified on any Remote System that is to participate in remote or distributed testing. In this context, the Remote Systems are other systems on which test cases or test case parts are run other than the Local System. **The Local System always has a system ID of zero (000)**; other system IDs always refer to Remote Systems. In this context, the Local System is defined as the one on which TETware *professional* is invoked. Usually, different TETware *professional* systems are located on different physical machines. However, there is no reason why a single machine should not have several different system IDs.

Configuration variable assignments made on the Local System are propagated to each of the Remote Systems; however, configuration variable assignments made on a Remote System normally have precedence over those that are propagated from the Local System.

6.1 Adding a Remote System

Please note that adding a Remote System is only appropriate if you are using Distributed TETware *professional*.

Once TETware *professional* has been loading and you have already created or selected a `Test Run`, then you are able to add a `Remote System` by any of the 3 options listed below:

- Click on the icon to create a Remote System using the `Remote System Wizard`
- Use the `Add Remote System` menu item in the `Config` menu at the tool bar
- Right Click on the mouse while on the `Configuration Tree Node` and select `Add Remote System`.

If you click on the icon to create a Remote System, you will need to follow the on screen information and set up all the necessary variables that are requested, (some of these fields are mandatory while others are not).

6.1.1 Using the Remote Wizard to Add a Remote System

If you decided to use this method, you will be taken through a series of dialog boxes prompting you to enter configuration information. To do this effectively you will need to refer to Section 6.2 ‘Configuration setting on the Remote System’.

6.1.2 Using the Tree Node or Menu Bar to Add a Remote System

If using the menu bar then click on `Config`, point and click on `Add Remote System`. Or right click on the mouse while your cursor is over `Configuration` in the tree node and select `Add Remote System`.

1. A dialog box will appear with the message ‘`Please enter a name for the Remote System`’. It also reminds you to note that the Remote System name must be a fully qualified domain name or IP address. After entering the name or IP address, please click on `OK` to continue or `Cancel` to abort.

2. If you click on `OK`, without entering the information required, you will be prompted by an error message ‘`No Remote System name given or specified’`. If details are entered then it will prompt you to enter a Remote System number between 1 and 999.

3. If number 0 is entered here it will give an error message: ‘`Error: System number 000 is reserved for the Local System`’ or if a number greater than 999 was entered, it returns an error message ‘Remote System number is invalid’. When a valid number is given, a Remote System will be created.

6.2 Configuration Settings on the Remote System

Configuration settings on Remote Systems are similar to those on the Local System. There are 4 tabs and they are:

- `Remote: Environment` tab
- `Remote: Build Mode field` tab
- `Remote: Execute Mode` tab
- `Remote: Clean Mode` tab

6.2.1 Remote: Environment

This tab allows you to specify variables for the operation of TETware *professional* on the Remote System concerned (please refer to Figure 16).

`TET_ROOT` This variable specifies to TETware *professional* the location of library files and certain configuration files that must exist under the `TET_ROOT` directory for the Remote System. This variable MUST always be specified for a Remote System participating in testing.

`Mapping` This specifies the mapped path of the TET_ROOT directory for the Remote System.

Note: It may be that the Remote System to be used in testing is not accessible via the file system, i.e. the remote file system is not mounted or no network drive has been mapped, in which case the directory to use should be typed directly into the text field. Use the `Clear` button to clear the contents of the relevant field.

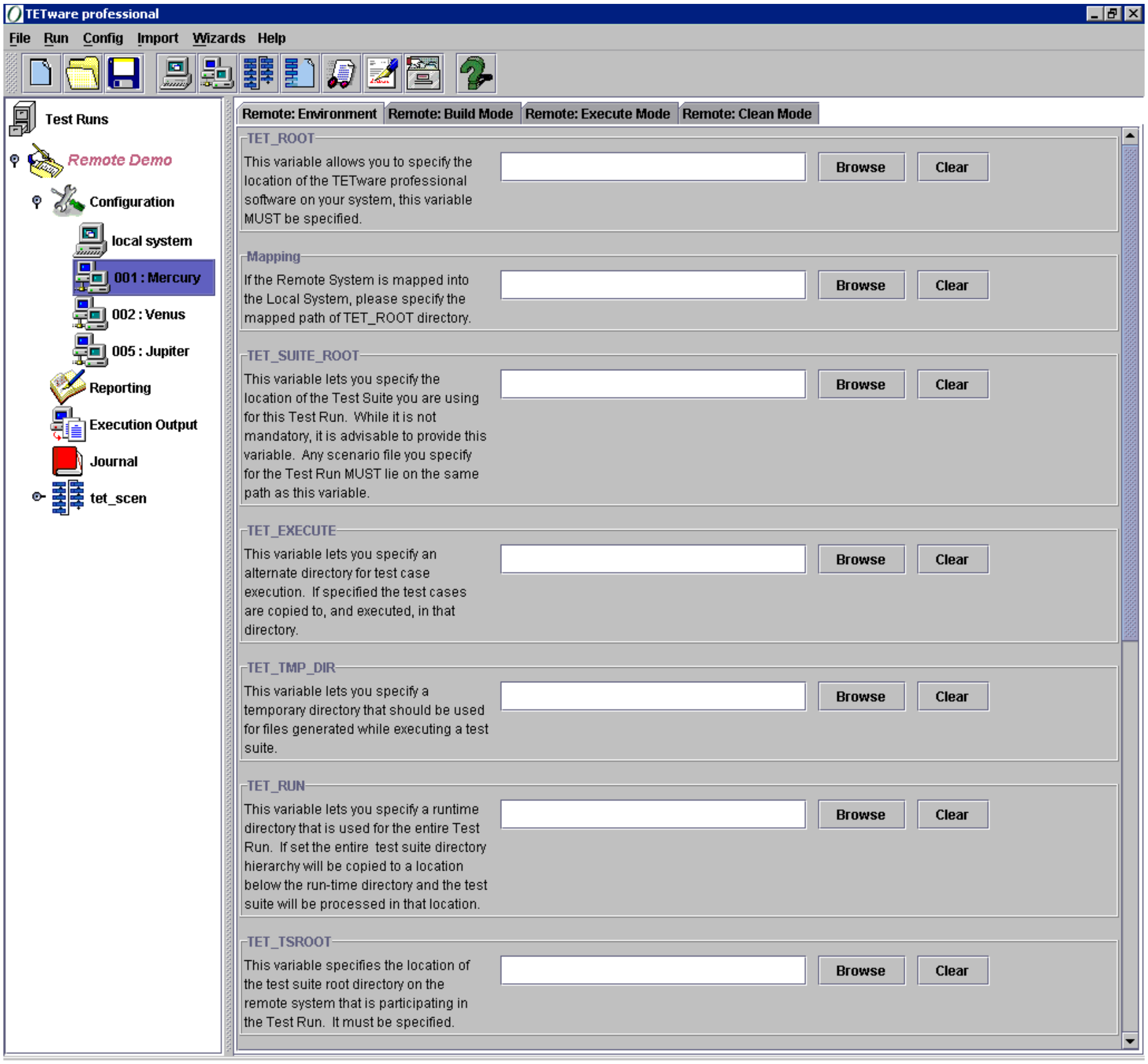
`TET_SUITE_ROOT` For Remote Systems this variable is not used by TETware *professional* but if specified is passed to Test Cases and tools in the environment as a communication variable.

`TET_EXECUTE` If specified this variable indicates to TETware *professional* that it should perform Test Case execution in the directory specified on the Remote System. This variable is passed as a communication variable.

`TET_TMP_DIR` If specified this variable indicates to TETware *professional* that it should place all files generated by Test Case activity under the directory specified. This variable is passed as a communication variable.

`TET_RUN` If specified this variable indicates to TETware *professional* that it should copy the entire Test Suite to the directory specified and process the Test Suite in that directory. This variable is passed as a communication variable.

`TET_TSROOT` This variable is only found on Remote Systems and is used to indicate the location of the Test Suite root directory on the Remote System. **It must be specified for each Remote System participating in the Test Run.**



TET_ROOT

This variable allows you to specify the location of the TETware professional software on your system, this variable MUST be specified.

Mapping

If the Remote System is mapped into the Local System, please specify the mapped path of TET_ROOT directory.

TET_SUITE_ROOT

This variable lets you specify the location of the Test Suite you are using for this Test Run. While it is not mandatory, it is advisable to provide this variable. Any scenario file you specify for the Test Run MUST lie on the same path as this variable.

TET_EXECUTE

This variable lets you specify an alternate directory for test case execution. If specified the test cases are copied to, and executed, in that directory.

TET_TMP_DIR

This variable lets you specify a temporary directory that should be used for files generated while executing a test suite.

TET_RUN

This variable lets you specify a runtime directory that is used for the entire Test Run. If set the entire test suite directory hierarchy will be copied to a location below the run-time directory and the test suite will be processed in that location.

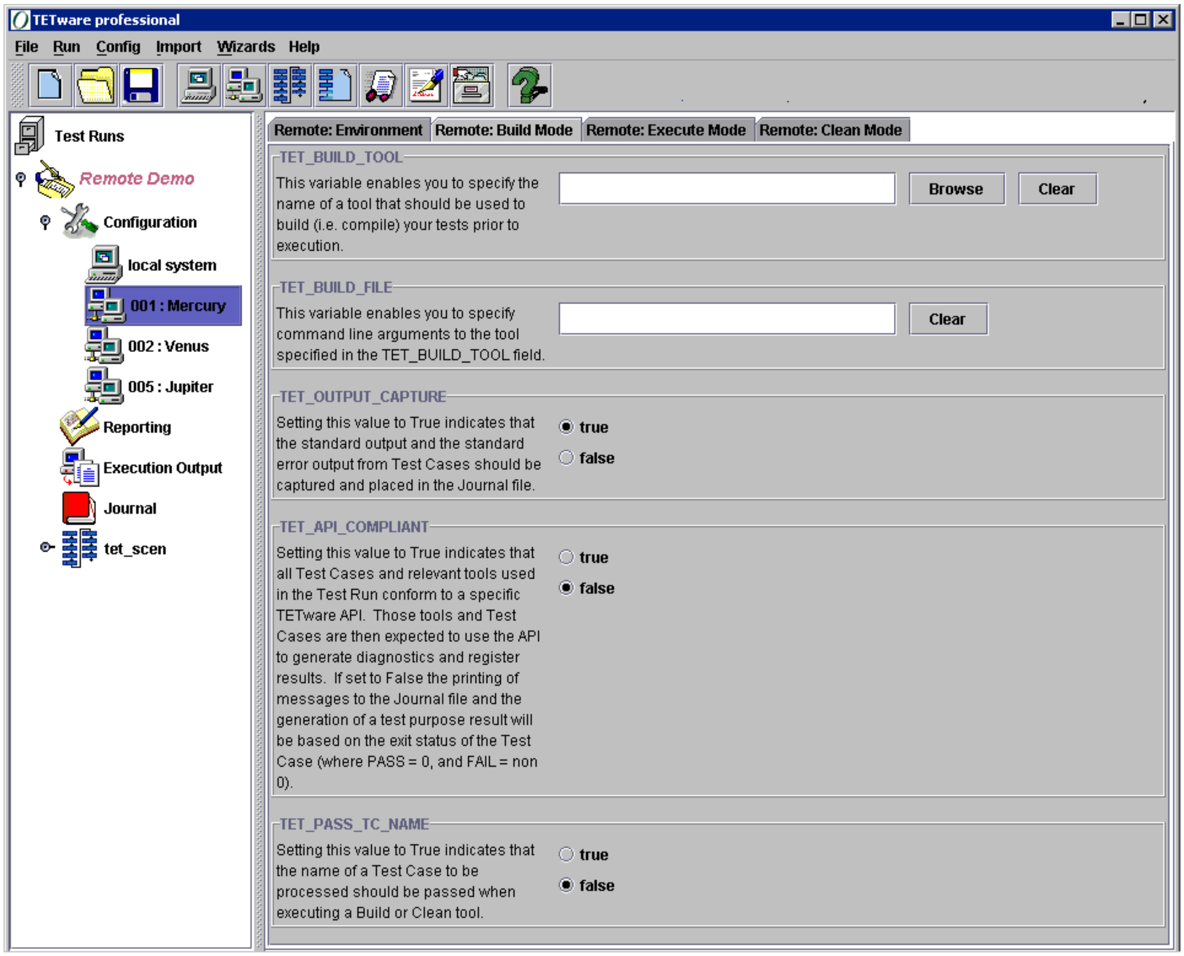
TET_TSROOT

This variable specifies the location of the the test suite root directory on the remote system that is participating in the Test Run. It must be specified.

Figure 16: Remote: Environment

6.2.2 Remote: Build Mode

This tab allows you to specify variables that will be used by TETware *professional* in Build mode for the Remote System specified. (Please see Figure 17).



TET_BUILD_TOOL

This variable enables you to specify the name of a tool that should be used to build (i.e. compile) your tests prior to execution.

TET_BUILD_FILE

This variable enables you to specify command line arguments to the tool specified in the TET_BUILD_TOOL field.

TET_OUTPUT_CAPTURE

Setting this value to True indicates that the standard output and the standard error output from Test Cases should be captured and placed in the Journal file.

● true
○ false

TET_API_COMPLIANT

Setting this value to True indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to False the printing of messages to the Journal file and the generation of a test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

○ true
● false

TET_PASS_TC_NAME

Setting this value to True indicates that the name of a Test Case to be processed should be passed when executing a Build or Clean tool.

○ true
● false

Figure 17: Remote: Build Mode

`TET_BUILD_TOOL` This variable enables you to specify a tool that TETware *professional* should use to build test cases, for example ‘`make`’.

`TET_BUILD_FILE` This variable allows you to specify additional command line arguments that can be passed to the build tool specified by `TET_BUILD_TOOL`. Please note: no validation is performed by TETware *professional* for the relationship between `TET_BUILD_TOOL` and `TET_BUILD_FILE`, it is the Users responsibility to ensure that they interact correctly.

`TET_OUTPUT_CAPTURE` These radio buttons indicate to TETware *professional* whether the standard output and error streams from the Build tool should be captured and placed in the Journal file. The default value of this variable is `True`. Setting this value to `False` indicates that the standard output from Test Cases should not be captured and placed in the Journal file.

TET_API_COMPLIANT These radio buttons allows you to specify to TETware *professional* whether the Build tool and Test Cases to be built conform to TETware *professional* API, i.e. they use TETware *professional* API to output information to the Journal file. The default value of this variable is `False`. Setting the value to `True` indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to `False` the printing of messages to the journal file and the generation of the test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

TET_PASS_TC_NAME These radio buttons variable allows you to specify to TETware *professional* whether the name of the current Test Case should be passed to the Build tool specified. The default value of this variable is `False`. Setting this value to `True` indicates that the name of the Test Case to be processed should be passed when executing a Build or Clean tool.

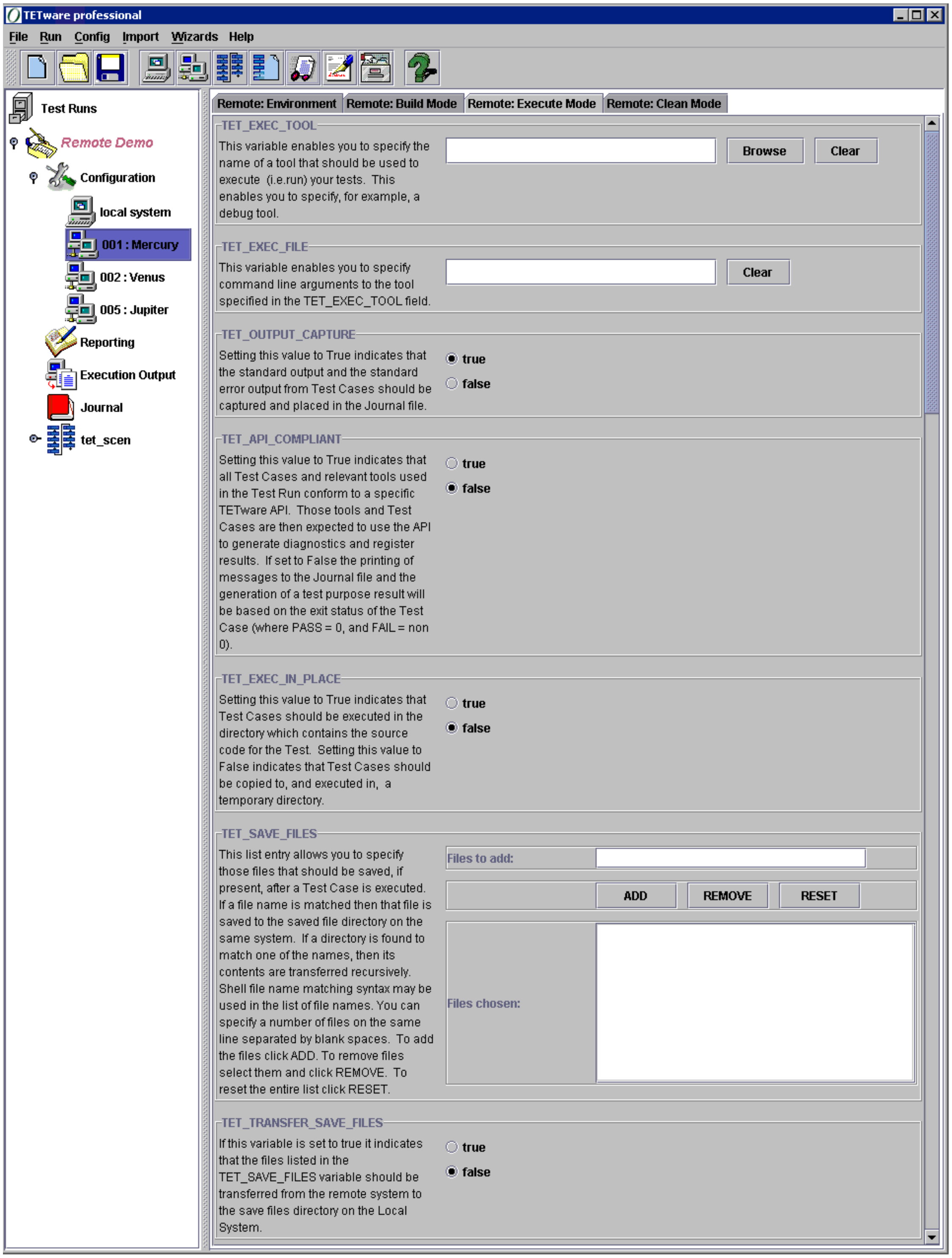
6.2.3 Remote: Execute Mode

This tab allows you to specify variables that will be used by TETware *professional* in Execute mode. (Please see Figure 18).

TET_EXEC_TOOL This variable enables you to specify a tool that TETware *professional* should use to execute test cases.

TET_EXEC_FILE This variable allows you to specify additional command line arguments that can be passed to the execute tool specified by `TET_EXEC_TOOL`. Please note: no validation is performed by TETware *professional* for the relationship between `TET_EXEC_TOOL` and `TET_EXEC_FILE`, it is the Users responsibility to ensure that they interact correctly.

TET_OUTPUT_CAPTURE These radio buttons indicate to TETware *professional* whether the standard output and error streams from the Execute tool should be captured and placed in the Journal file. The default value of this variable is `True`. Setting this value to `False` indicates that the standard output and the standard error output from Test Cases should not be captured and placed in the Journal file.



TET_EXEC_TOOL

This variable enables you to specify the name of a tool that should be used to execute (i.e.run) your tests. This enables you to specify, for example, a debug tool.

TET_EXEC_FILE

This variable enables you to specify command line arguments to the tool specified in the TET_EXEC_TOOL field.

TET_OUTPUT_CAPTURE

Setting this value to True indicates that the standard output and the standard error output from Test Cases should be captured and placed in the Journal file.

● true
○ false

TET_API_COMPLIANT

Setting this value to True indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to False the printing of messages to the Journal file and the generation of a test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

○ true
● false

TET_EXEC_IN_PLACE

Setting this value to True indicates that Test Cases should be executed in the directory which contains the source code for the Test. Setting this value to False indicates that Test Cases should be copied to, and executed in, a temporary directory.

○ true
● false

TET_SAVE_FILES

This list entry allows you to specify those files that should be saved, if present, after a Test Case is executed. If a file name is matched then that file is saved to the saved file directory on the same system. If a directory is found to match one of the names, then its contents are transferred recursively. Shell file name matching syntax may be used in the list of file names. You can specify a number of files on the same line separated by blank spaces. To add the files click ADD. To remove files select them and click REMOVE. To reset the entire list click RESET.

TET_TRANSFER_SAVE_FILES

If this variable is set to true it indicates that the files listed in the TET_SAVE_FILES variable should be transferred from the remote system to the save files directory on the Local System.

○ true
● false

Figure 18: Remote: Execute Mode

TET_API_COMPLIANT These radio buttons allow you to specify to TETware *professional* whether the Execute tool and Test Cases to be built conform to TETware *professional* API, i.e. they use TETware *professional* API to output information to the Journal file. The default value of this variable is `False`. Setting this value to `True` indicated that all Test Cases and relevant tool used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to `False` the printing of the Journal file and the generation of a Test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

TET_EXEC_IN_PLACE These radio buttons indicate to TETware *professional* that it should execute Test Cases in the directory that contains the source code for the Tests. Setting this value to `True` indicates that Test cases should be executed in the directory that contains the source code for the Test. Setting the value to `False` indicates that the Test Case should be copied to and executed to and executed in a temporary directory.

TET_SAVE_FILES You can specify a number of files that TETware *professional* should save after Test execution (if it is able to find those files). You can specify directories or normal shell matching syntax (i.e. *) as part of the list. To add the files enter them in the '`Files to Add`' text field and then press ADD.

To remove files, select the rows in the list and press RESET, you can select a contiguous list of files by pressing and holding '`Shift`' whilst clicking the files. You can select a dis-contiguous list of files by pressing and holding `'Ctrl`' whilst clicking the files.

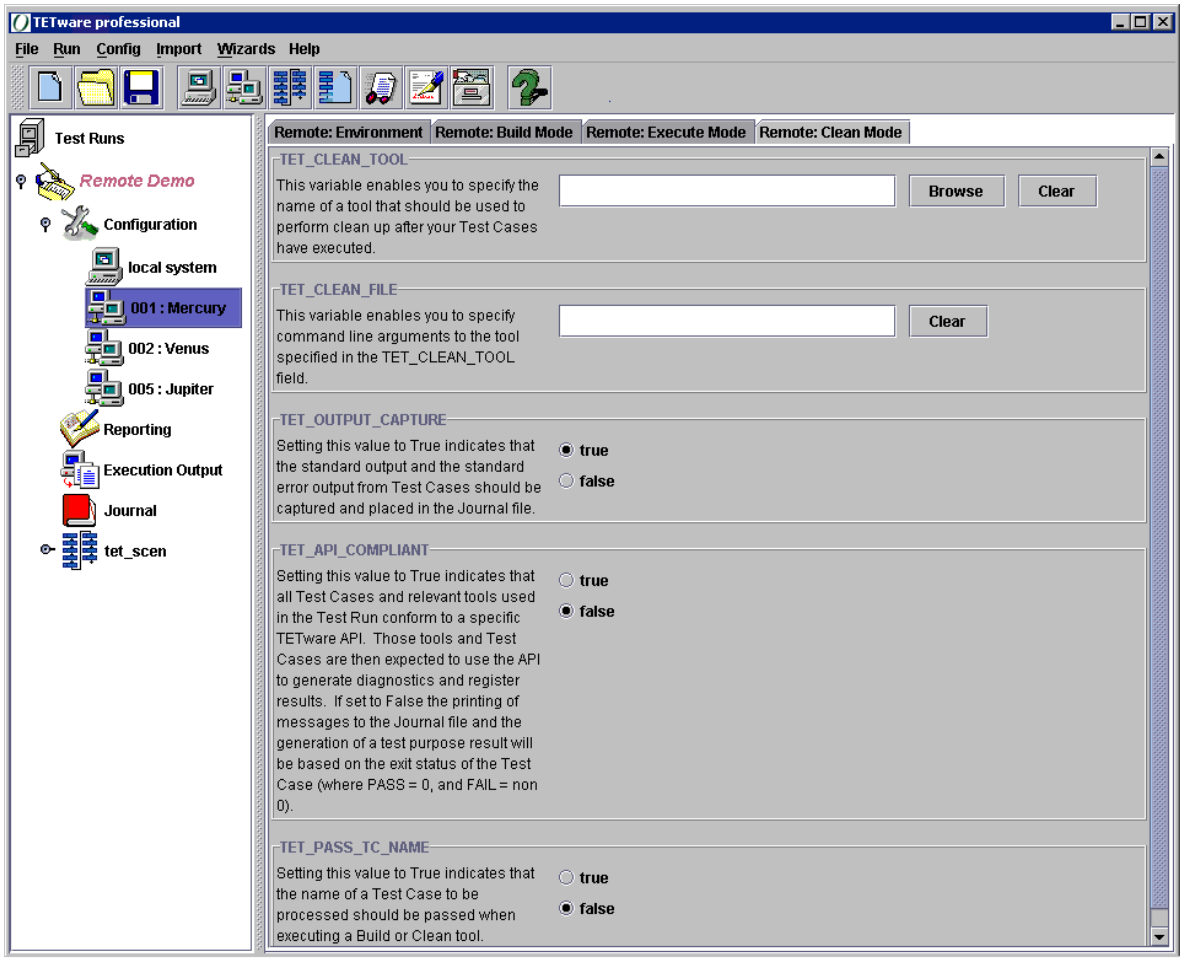
To remove all entries in the list of files, click RESET.

TET_TRANSFER_SAVE_FILES This radio button variable indicates to TETware *professional* that it should transfer the files specified in the TET_SAVE_FILES variable back to the Local System instead of being saved on the Remote System. The default value of this variable is `False`. Setting this value to `True` indicates that the files listed on this variable should be transferred from the remote system to the save files directory on the local system.

Note: these variables will be given a suffix of 'TET_REM*nnn*' when written to the relevant configuration file, where nnn is the Remote System number assigned by the User.

6.2.4 Remote: Clean Mode

This tab allows you to specify variables that will be used by TETware *professional* in Clean mode.



TET_CLEAN_TOOL

This variable enables you to specify the name of a tool that should be used to perform clean up after your Test Cases have executed.

TET_CLEAN_FILE

This variable enables you to specify command line arguments to the tool specified in the TET_CLEAN_TOOL field.

TET_OUTPUT_CAPTURE

Setting this value to True indicates that the standard output and the standard error output from Test Cases should be captured and placed in the Journal file.

● true
○ false

TET_API_COMPLIANT

Setting this value to True indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to False the printing of messages to the Journal file and the generation of a test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

○ true
● false

TET_PASS_TC_NAME

Setting this value to True indicates that the name of a Test Case to be processed should be passed when executing a Build or Clean tool.

○ true
● false

Figure 19: Remote: Clean Mode

`TET_CLEAN_TOOL` This variable enables you to specify a tool that TETware *professional* should use to clean test cases.

TET_CLEAN_FILE This variable allows you to specify additional command line arguments that can be passed to the clean tool specified by TET_CLEAN_TOOL. Please note: no validation is performed by TETware *professional* for the relationship between TET_CLEAN_TOOL and TET_CLEAN_FILE, it is the Users responsibility to ensure that they interact correctly.

TET_OUTPUT_CAPTURE These radio buttons indicate to TETware *professional* whether the standard output and error streams from the Clean tool should be captured and placed in the Journal file. The default value of this variable is True. Setting this value to False indicates that the standard output and the standard error output from Test Cases should not be captured and placed in the journal file.

TET_API_COMPLIANT These radio buttons allows you to specify whether the Clean tool and Test Cases to be built conform to TETware *professional* API, i.e. they use TETware *professional* API to output information to the Journal file. The default value of this variable is False. Setting the value to TRUE indicates that all Test Cases and relevant tools used in the Test Run conform to a specific TETware API. Those tools and Test Cases are then expected to use the API to generate diagnostics and register results. If set to FALSE the printing of messages to the journal file and the generation of the test purpose result will be based on the exit status of the Test Case (where PASS = 0, and FAIL = non 0).

TET_PASS_TC_NAME These radio buttons allow you to specify to TETware *professional* whether the name of the current Test Case should be passed to the Clean tool specified. The default value of this variable is FALSE. Setting this value to TRUE indicates that the name of a Test Case to be processed should be passed when executing a Build or Clean tool.

Note: these variables will be given a suffix of '*TET_REMnnn*' when written to the relevant configuration file, where *nnn* is the Remote System number assigned by the User.

6.3 Importing Configuration Files

TETware *professional* 1.4 allows the import of build, execute and clean mode configuration files to the relevant tab and fields for the configuration of the Remote System highlighted in the tree. This feature makes it easier to configure a Remote System from local machine.

6.3.1 Importing Build Mode File

There are two ways to import a build mode configuration file:

1. When the `Remote System` is highlighted, select from the Menu: `Import / Import Build mode file`, a File Chooser will be shown. Select the file you want to import to the Remote System, then click on the `Open` button, the configuration value will appear in the relevant field under the `Remote: Build Mode` tab.

2. Right-click the `Remote System` tree node, a pop-up menu will be shown. Select `Import Build mode file`, then follow the same procedure as above.

6.3.2 Importing Execute Mode File

There are two ways to import a execute mode configuration file:

1. When the `Remote System` is highlighted, select from the Menu: `Import / Import Execute mode file`, a File Chooser will be shown. Select the file you want to import to the Remote System, then click on the `Open` button, the configuration value will appear in the relevant field under the `Remote: Execute Mode` tab.

2. Right-click the `Remote System` tree node, a pop-up menu will be shown. Select `Import Execute mode file`, then follow the same procedure as above.

6.3.3 Importing Clean Mode File

There are two ways to import a clean mode configuration file:

1. When the `Remote System` is highlighted, select from the Menu: `Import / Import Clean mode file`, a File Chooser will be shown. Select the file you want to import to the Remote System, then click on the `Open` button, the configuration value will appear in the relevant field under the `Remote: Clean Mode` tab.

2. Right-click the `Remote System` tree node, a pop-up menu will be shown. Select `Import Clean mode file`, then follow the same procedure as above.

6.4 Copying a Remote System

The advantage that this feature offers is that it helps to saves time as it copies all of the configuration settings of one Remote System to a new Remote System.

To Copy a Remote System, you will need to have created at least one Remote System first. You will then have to carry out the following steps

1. Right click on a previously created and configured Remote System.

2. Select the option `Copy Remote System`. A dialog box will come up giving you the option of entering a name for the new Remote System.

3. Enter the name of the new Remote System into the window, and then click on `OK`. If you click on `OK`, without entering the information required, you will be prompted by an error message '`No Remote System name given or specified'`.

4. If details are entered then you will be prompted to enter a Remote System number between 1 and 999. Enter the Remote System Number and click on `OK` to complete the Remote System copying. If number 0 is entered here it will give an error message: 'Error: System number 000 is reserved for the Local System' or if a number greater than 999 was entered, it returns an error message 'Remote System number is invalid'.

6.5 Removing a Remote System

There are two ways to remove a Remote System:

1. On the Remote System tree node, Right-click on the node, a pop-up menu will appear, select '`Remove Remote System`'.

2. Press the Delete key when the Remote System is highlighted.

You will then be prompted with a dialog box with the confirmation message: `Confirm removal of Remote System [`*`Number`*`]: [`*`Name`*`]`?

If the `Yes` button is clicked then all components of the Remote System chosen will be removed from the tree. If `No` is chosen then the Remote System will not be removed.

6.6 Rename Remote System

You can rename and/or renumber a Remote System by following the procedure below:

1. Right-click on a Remote System in the Test Run tree, a pop-up menu will be shown, one of the items is: Rename Remote System.

2. Select this menu item, a dialog box for entering new name will be displayed. Enter the name and click on the `OK` button. A new dialog box for entering the new Remote System number will be shown.

3. In the dialog box for the new number, enter the new system ID for the Remote System, and click on the `OK` button. A new dialog box to confirm the change will be shown. If the new number is invalid, for example, out of range of the Remote System numbers, an error message will be displayed to inform the user, and promote user to try again.

4. Once the confirmation is done, the new name and new number will replace the old name and old number of the Remote System, and the change will be reflected by the tree node label.

At any stage of the dialog box, the `Cancel` option will abort the operation.

6.7 Systems Files

These files are only used by Distributed TETware *professional*. They are not used by the Lite version.

6.7.1 The `Systems` File

The systems file should be present in the TET_ROOT directory on each system and is used by Distributed TETware *professional* to map TETware *professional* system IDs to some information that can be used by the network interface to identify a machine.

An example systems file is supplied with the demonstration in *`tet-root`*`/src/tet3/demo/systems`, which contains lines similar to the following:

```
# Example system file for demonstration
000 Local
001 Remote
```

The systems file is automatically generated in the TET_ROOT directory of the Local System when a Test Run is starting to execute. For Remote Systems, the file will be automatically copied to the mapped TET_ROOT directory of the Remote System, if the `Mapping` variable has been defined under the Remote: Environment tab. Otherwise, TETware *professional* will tell the user to manually copy the file to the right directory before the test run execution continues.

6.7.2 Makefiles

You must check the test case `makefiles` on each system to ensure that they will build the test cases correctly.

You may need to add the names of libraries to the `SYSLIBS` variable in order to resolve external function names used in the TETware *professional* and API. For example, if you have built TETware *professional* to use the socket network interface, you may need to append `-lsocket` and `-lnsl` to this variable on some systems.

If you are running either part of the distributed demonstration on a Windows NT System, you will need to customize the file name suffix variables in the `makefiles` as well as the `SYSLIBS` variable. Details of the changes that you must make are presented in comments contained in the `makefiles`.

7 Reporting – Configure Report Writer & Database

Before any report can be generated from the Test Run, the TETware Report Writer and/or Database need to be configured.

7.1 Configuring the TETware Report Writer

Depending on what information you require in your report, you can decide to check or uncheck the boxes next to the relevant variables (please see Figure 20). Some boxes are required while others are optional.

Report Generator:

This variable enables you to specify the location of TETware Report Writer. The default location is C:\tet_root\bin\ (where 'C' is the Drive letter, *tet_root* is your TETware *professional* root directory and 'bin' is the sub-directory)

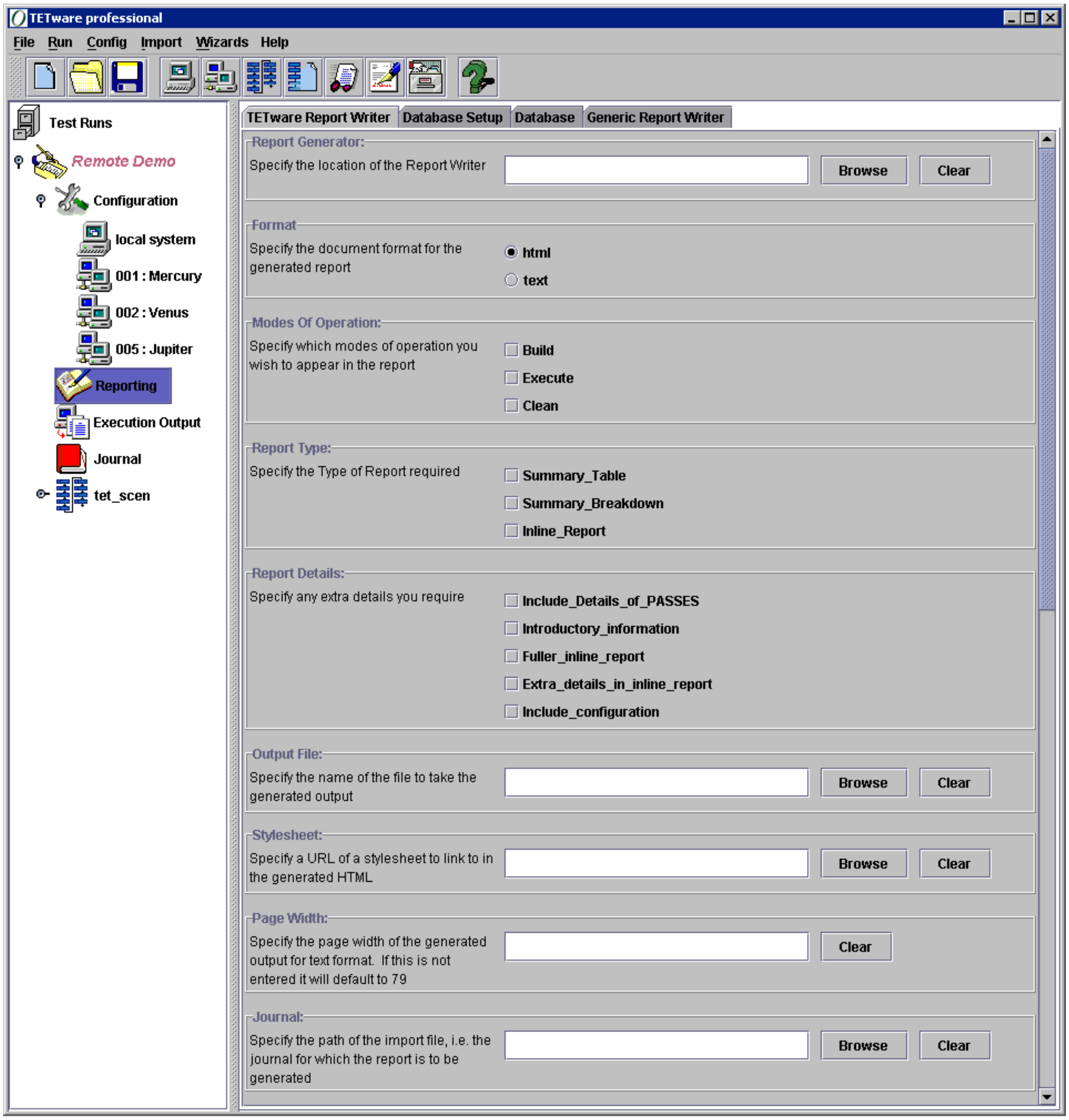
Format:

This specifies which document format you will want for the generated report. Either:

`Html`	Results can be viewed in HTML Format or
`Text`	Results can be viewed in test (txt) Format

Mode of Operation:

You will be able to specify which mode of operation should appear in the report. Any combination of the following can be chosen: Build, Execute or Clean. For more information on Modes of Operation please see Section 2.5.



Report Generator:

Specify the location of the Report Writer

Format

Specify the document format for the generated report

● html

○ text

Modes Of Operation:

Specify which modes of operation you wish to appear in the report

□ Build

□ Execute

□ Clean

Report Type:

Specify the Type of Report required

□ Summary_Table

□ Summary_Breakdown

□ Inline_Report

Report Details:

Specify any extra details you require

□ Include_Details_of_PASSES

□ Introductory_information

□ Fuller_inline_report

□ Extra_details_in_inline_report

□ Include_configuration

Output File:

Specify the name of the file to take the generated output

Stylesheet:

Specify a URL of a stylesheet to link to in the generated HTML

Page Width:

Specify the page width of the generated output for text format. If this is not entered it will default to 79

Journal:

Specify the path of the import file, i.e. the journal for which the report is to be generated

Figure 20: TETware Report Writer

Report Type:

This allows the user to specify the type(s) of report required. User can pick a combination of the following: -

Summary_table Provides a summary of the results by mode of operation.

Summary_breakdown Provides a summary of the results by mode of operation and test case.

Inline_report. Provides a summary of the results by test case and mode of operation.

Report Details:

This allow user to specify any extra detail is required. Such details include:

Include_Details of PASSES If this option is selected details of tests that PASS will be included. Not selecting this item can help locate problems.

Introductory_information This adds the: TETware version; System information; Date of Test Run and Start time to the top of your Report.

Fuller_inline_Report This is an Inline Report broken down by Test Cases and Test Purpose (TP). You must have selected Report Type: Inline_Report for this option to be available.

Extra_Details_in_inline_report This adds Scenario information to the Inline Report. You must have selected Report Type: Inline_Report for this option to be available.

Include_configuration This adds configuration information, by variable and value, to the top of the report (after introductory information).

Output File:

This specifies the name of the file to which you want the generated output to be written. (If this file is already in existence, then you can click on the `Browse` button to select the file). You must specify the path and the suffix as well as the file name.

Stylesheet:

This option or field applies only if the html format has been selected. If specified the URL of the stylesheet to link to in the generated HTML.

Page Width:

This option applies only to text format reports. It specifies the page width of the generated output. If this is not entered, it will use the default - which is 79 – will be used.

Journal:

This specifies the path of the import file, i.e. the journal from which the report is to be generated (This file can also be browsed and selected). After a Test Run has been executed this field is automatically completed. If the Test Run is saved the Journal name that was current when the Test Run is saved is also saved.

7.1.1 Creating a Report

Select the Run TETware Report Writer button on the toolbar.

You can then go and view your report at the appropriate location using a tool that is appropriate for the format selected, e.g. your browser if the Report is saved in HTML format.

7.2 Configuring the Database Extraction tool

To be able to extract journal file parameters to the Database, some variables needs to be set on the Database Setup and Database tabs.

7.2.1 Database Setup tab

For details of the Administrative aspects of Database Setup please see Section 2.4 of the TETware *professional* Installation Guide.

TETware *professional* Database Setup on Windows, UNIX and Linux Platform are all the same. It can be varied each time a Test Run is configured.

7.2.1.1 Database Connection:

This refers to the Database connection being used. Consideration is given to the way in which TETware *professional* communicates with the users Database.

In Figure 21 `jdbc:odbc:`*`TETbase`* was used to configure a connection to a Microsoft Access database. Some other examples that could be used depending on the Database the individual is connecting to includes:

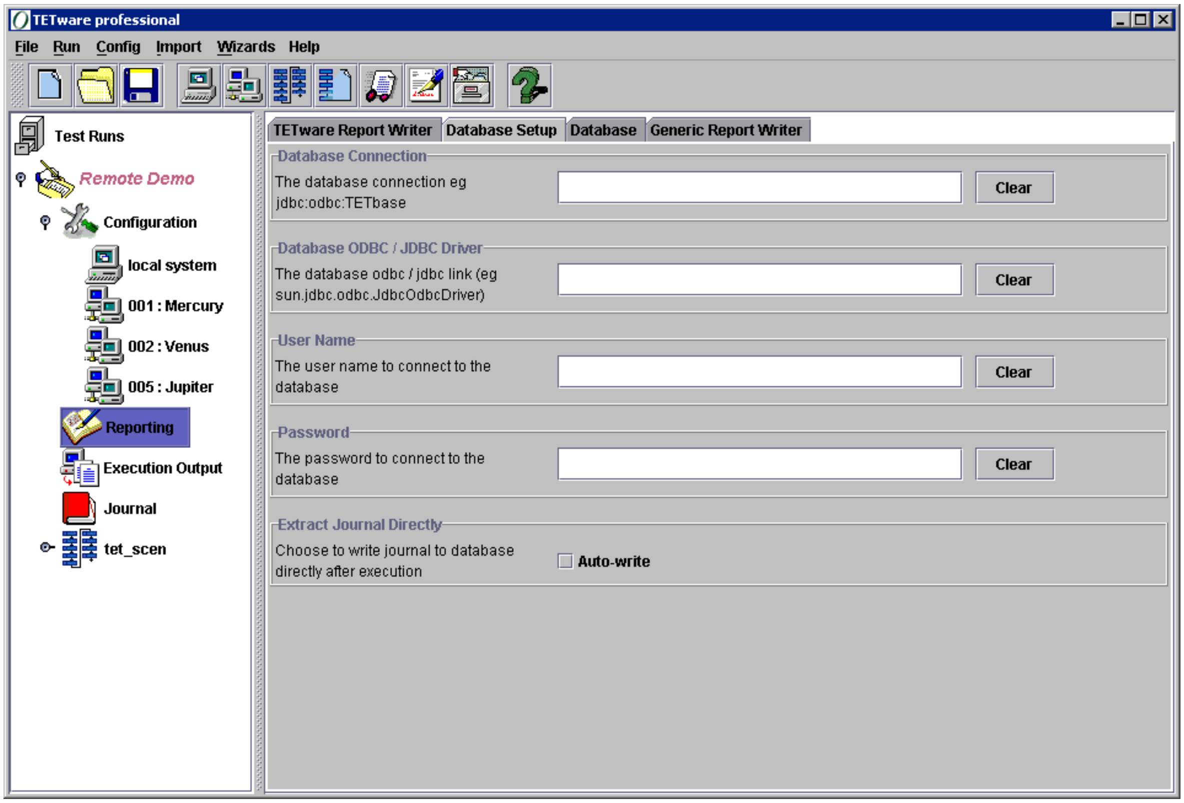
MYSQL	`jdbc:mysql://localhost/`*`TETbase`*
ORACLE	`jdbc:oracle:thin@localhost:1521:`*`TETbase`*
INFORMIX	`jdbc:Informix:apdirect:system=tetsystem=`*`TETbase`* etc.

7.2.1.2 Database ODBC / JDBC Driver:

This field refers to the Database Links that are being used. In Figure 21, we have used the Open Database Connectivity (ODBC) and Java Database Connectivity (JDBC) Drivers `Sun.jdbc.odbc.jdbcOdbcDriver` to configure a connection to a Microsoft Access database. Some other examples that may be used include:

MYSQL	`org.gjt.mm.mysql.Driver`

ORACLE	`oracle.jdbc.driver.OracleDriver`
INFORMIX	`Informix.api.jdbc.JDBCDriver` *etc.*



Database Connection
The database connection eg jdbc:odbc:TETbase

Database ODBC / JDBC Driver
The database odbc / jdbc link (eg sun.jdbc.odbc.JdbcOdbcDriver)

User Name
The user name to connect to the database

Password
The password to connect to the database

Extract Journal Directly
Choose to write journal to database directly after execution
□ Auto-write

Figure 21: Database Setup tab

7.2.2 Remote Database:

If you are connected to a Database on a different system, you may be required by that Database system's security to supply a '`User's Name`' and a '`Password`'. This can be entered here so that it is not required every time a data connection is made.

7.2.2.1 User Name:

If your system requires a users name to access your Database file, then this users name has to be entered into this field.

7.2.2.2 Password:

If your system requires a password to access your Database file, then this password has to be entered into this field.

7.2.3 Extract Journal Directly

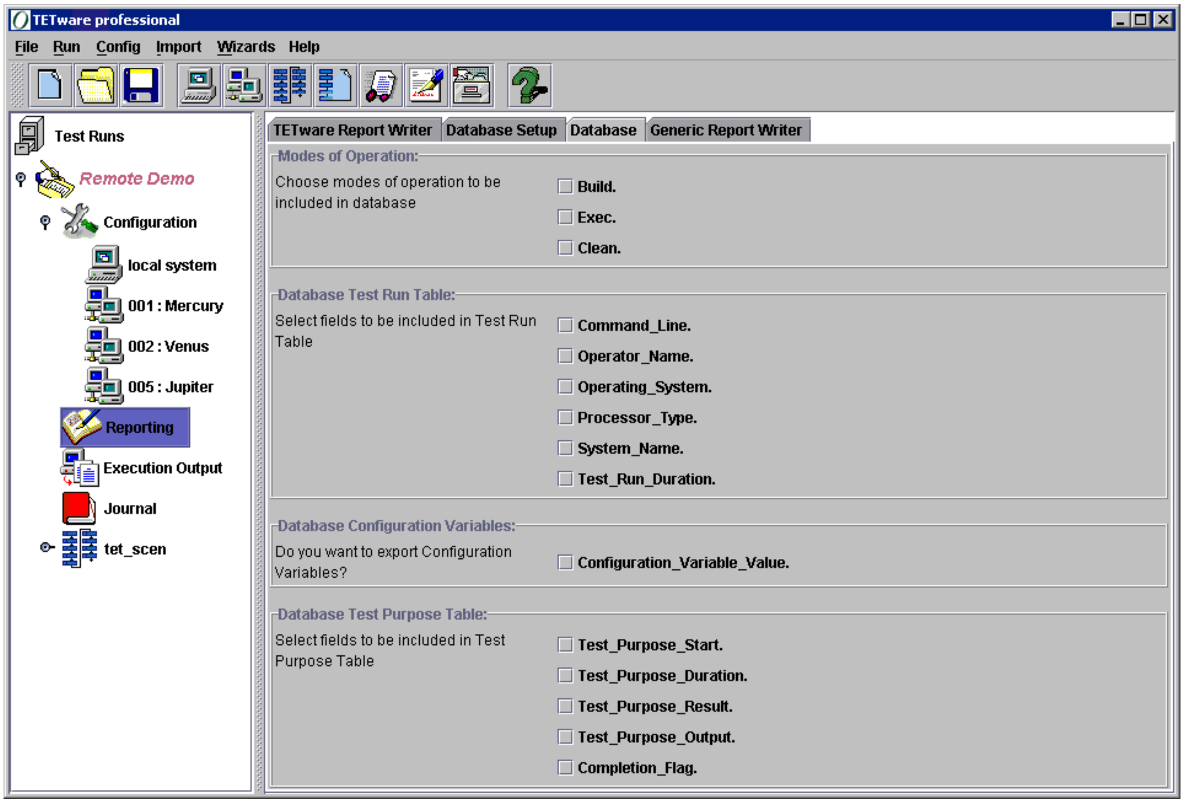
If you want the journal file to be extracted to the database **automatically** after the test run execution is complete, then you should check the `Auto-write` checkbox.

7.2.4 Database

Once the Database Setup is completed the user must select the `Database` tab, within the Reporting tree node, and then select the journal file parameters that they wish to pass to their database. These parameters are grouped according to the destination table in the database (please see Appendix B). Note that not all parameters or database fields are listed here. The key fields in each database table are mandatory and the relevant journal file parameters will be written to these fields each time the TETware *professional* Database Extract tool is run.

7.2.4.1 Mode of Operation:

You will need to select the modes of operation for which you wish to pass Journal File parameters to the database. The options available are `Build`, `Execute` and `Clean`.



Modes of Operation:

Choose modes of operation to be included in database

- □ Build.
- □ Exec.
- □ Clean.

Database Test Run Table:

Select fields to be included in Test Run Table

- □ Command_Line.
- □ Operator_Name.
- □ Operating_System.
- □ Processor_Type.
- □ System_Name.
- □ Test_Run_Duration.

Database Configuration Variables:

Do you want to export Configuration Variables?

- □ Configuration_Variable_Value.

Database Test Purpose Table:

Select fields to be included in Test Purpose Table

- □ Test_Purpose_Start.
- □ Test_Purpose_Duration.
- □ Test_Purpose_Result.
- □ Test_Purpose_Output.
- □ Completion_Flag.

Figure 22: Database Tab

One or more of these options must be selected if you want to pass results to the `Test Purpose` table. If no Mode of Operation is selected the relevant parameters will only be passed to the `Test Run` and `Configuration Variables` tables.

7.2.4.2 Database Test Run Table:

This option allow users to select fields to be included in Test Run Table. Available options are:

Command Line:	The command line used to execute the test.
Operators Name:	The name of the person who is logged on for this Test Run
Operating System:	The operating system resident on the system.
Processor Type:	The type of processor that the system is using.

System Name: The name of the system that is being used.

Test_Run_Duration: The time used to run the test

7.2.4.3 Database Configuration Variables:

Selecting these fields causes the configuration variables to be exported to the Configuration Variables table.

7.2.4.4 Database Test Purpose Table:

This option allows users to select fields to be included in Test Purpose Table. Available options are:

Test_Purpose_Start: The start time of the test purpose.

Test_Purpose_Duration: The time taken to complete the test purpose.

Test_Purpose_Result: The result code for this test purpose.

Test_Purpose_Output: Output from this invocation of the test purpose.

Completion_Flag: Flag indicating tests that failed to complete.

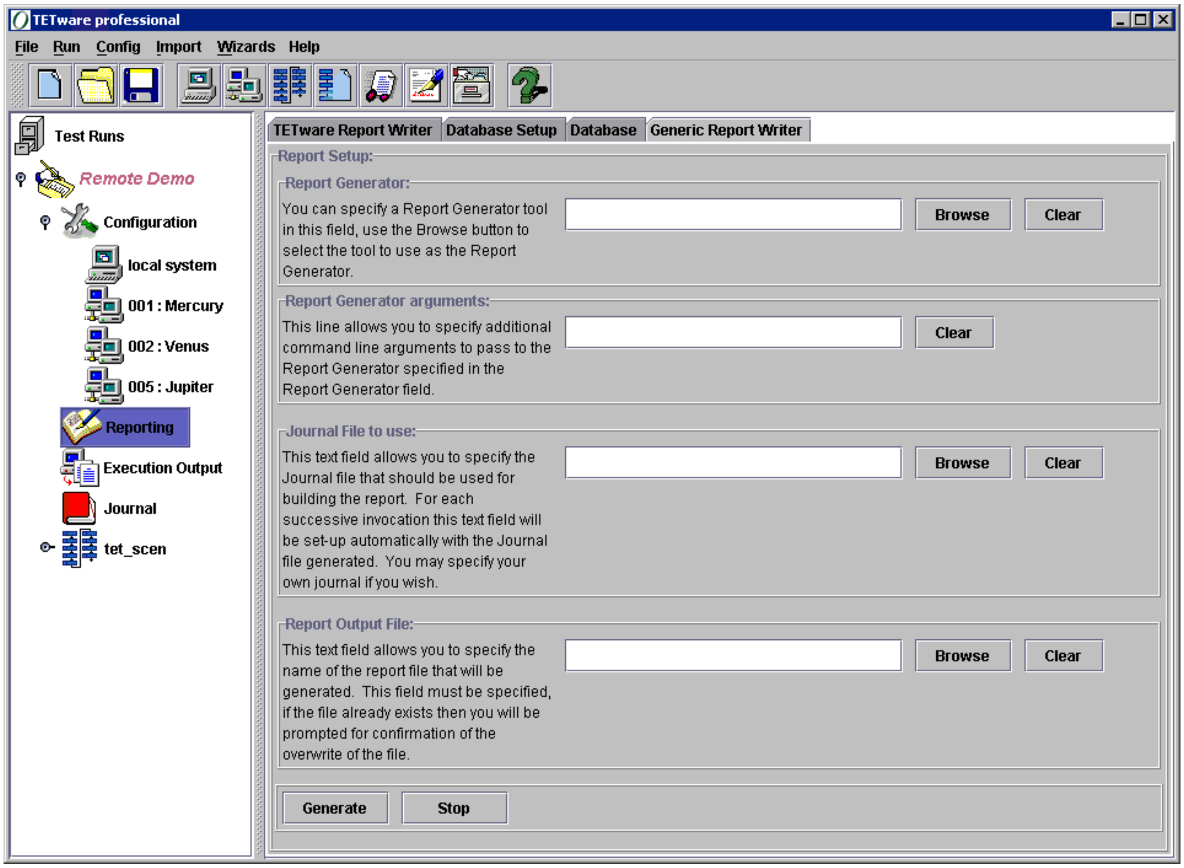
7.3 Configuring your own Report Writer

It is possible to generate reports for a Test Run using your own Report Writer. (Please see Figure 23).

The following criteria must be met for a report to be successfully generated for a Test Run:

Report Generator:

A valid Report Writer must be specified in this text field, the file system can be browsed to find the required Report Writer.



You can specify a Report Generator tool in this field, use the Browse button to select the tool to use as the Report Generator.

This line allows you to specify additional command line arguments to pass to the Report Generator specified in the Report Generator field.

This text field allows you to specify the Journal file that should be used for building the report. For each successive invocation this text field will be set-up automatically with the Journal file generated. You may specify your own journal if you wish.

This text field allows you to specify the name of the report file that will be generated. This field must be specified, if the file already exists then you will be prompted for confirmation of the overwrite of the file.

Figure 23: Generic Report Writer

Report Generator Argument:

The Report Writer must take the name of the Journal file as an argument.

Journal File to use:

The Report Writer must send it's output to the standard output stream. This is because the output is captured by TETware *professional* and written to the output file.

Report Output File:

An output file must be specified in the `Report Output File` text field. The file system can be browsed to find an existing file, if an existing file is used then you will be prompted to overwrite the file. If no output file is specified then Report Generator Failed dialog box is displayed: with the message `Error: The output file must be specified`.

8 Executing a Test Run

To start a Test Run, you need to either right-click on the icon representing Test Run or use the '`Execute Test Run`' menu item in the '`Run`' menu. There are some things that you need to be bear in mind first.

8.1 Pre-execution Criteria

To successfully execute a Test Run, a number of criteria that MUST be met, they are:

`TET_ROOT` must be specified for the Local System: Without `TET_ROOT` specified TETware *professional* is unable to find important library files and certain configuration files. See Section 4.1.1.

`TET_SUITE_ROOT` should be specified: The rules applicable to `TET_SUITE_ROOT` are as follows:

- If the path to the Scenario file lies on the same path as `TET_ROOT` then `TET_SUITE_ROOT` does NOT need to be specified.

- If the path to the Scenario file does NOT lie on the same path as `TET_ROOT` then `TET_SUITE_ROOT` MUST be specified. i.e. if the Scenario file path is `/home/testsuite/tests/scen1.scn` and `TET_ROOT` is `/home/tet`, then `TET_SUITE_ROOT` must be specified as `/home/testsuite`.

- If `TET_SUITE_ROOT` is specified then the Scenario file must lie on the same path. i.e. if `TET_SUITE_ROOT` is `/home/testsuite` then the path to the Scenario file must start with `/home/testsuite`.

For further details please see Section 4.1.1.

At least one mode of operation MUST be specified
TETware *professional* needs to know what you want to do before it starts working! (See Section 4.1.1).

All Environment variables for the Local System MUST be directories

The Environment variables specified for the Local System MUST be directories since they are locations where TETware *professional* will either look for files or place files. (See Section 4.2.1).

Alternate configuration files must exist

If the User specifies any alternative configuration files then they must already exist. (See Section 4.1.4).

If distributed testing is to be performed (i.e. Remote Systems were added to the Test Run), then the following criteria MUST be met for each Remote System:

> `TET_ROOT` must be specified for each Remote System (see Section 6.2.1).
>
> `TET_TSROOT` must be specified for each Remote System (see Section 6.2.1).

One or more scenarios MUST be selected

Testing cannot commence unless TETware *professional* knows what it should be testing, so at least one Scenario must be selected from the Scenario file (see Section 5.3).

8.2 Starting Execution of a Test Run

After a Test Run has been selected, a scenario file picked and Test Cases are assigned, then a Test Run can be executed.

This is simply done by either of the three ways listed below:

- Click on the icon representing the Test Run on the Toolbar
- Use the `Execute Test Run` menu item in the `Run` menu
- Use the Keyboard Shortcut Alt-E

Whichever way you choose to execute the `Test Run`, you will be prompted with a dialog which asks you to confirm start of execution for Test Run: [*Name of Test Run*].

> If the `Yes` button is clicked then TETware professional will perform all the necessary checks to ensure that the Test Run can be executed, and if these are passed the Test Run Execution will begin. If any of the criteria outlined above are NOT met by the configuration for the Test Run then an appropriate dialog box will be displayed indicating the error (the dialog boxes are not displayed here due to the large number possible).

If all the criteria outlined above ARE met by the configuration for the Test Run then TETware *professional* will then generate all the configuration files that are needed by TETware *professional* to perform a successful execution (please see Section 8.3).

If `No` is chosen then the Test Run will be aborted.

When a Test Run begins to execute, the Display Window will automatically switch to the Execution Output window and the tree icon will change to ⛔. Once execution is complete the tree icon will change back to 🖥.

8.3 Execution Configuration Files

If all the criteria outlined above ARE met by the configuration for the Test Run then TETware *professional* will then generate the following configuration files that are needed by TETware *professional* to perform a successful execution:

`_GUItetbuild.cfg`
This file is the build mode file for the Test Run, it is only generated if the User has selected Build mode as a mode of operation. The location of this file is dependent on the values of `TET_ROOT` and `TET_SUITE_ROOT`, if `TET_ROOT` is set then the file will be placed under it, if `TET_SUITE_ROOT` then it will be placed under there.

`_GUItetexec.cfg`
This file is the execute mode file for the Test Run, it is only generated if the User has selected Execute mode as a mode of operation. The location of this file is dependent on the values of `TET_ROOT` and `TET_SUITE_ROOT`, if `TET_ROOT` is set then the file will be placed under it, if `TET_SUITE_ROOT` then it will be placed under there.

`_GUItetclean.cfg`
This file is the clean mode file for the Test Run, it is only generated if the User has selected Clean mode as a mode of operation. The location of this file is dependent on the values of `TET_ROOT` and `TET_SUITE_ROOT`, if `TET_ROOT` is set then the file will be placed under it, if `TET_SUITE_ROOT` then it will be placed under there.

`_scenTETfile`

This is a scenario file that is generated by TETware *professional* based upon the scenarios chosen. The file contains a TETware *professional* generated scenario: `_TETallscen`, this scenario contains the list of scenarios chosen by the User. In this way the chosen scenarios can be executed by TETware *professional*. The location of this file is dependent on the values of `TET_ROOT` and `TET_SUITE_ROOT`, if `TET_ROOT` is set then the file will be placed under it, if `TET_SUITE_ROOT` then it will be placed under there.

Please note: this file is temporary and will be removed after the Test Run has finished execution, it is mentioned here for completeness.

`tet_code`

This file is only generated if the User specifies Custom Result Codes, (see Section 4.1.3). The location of this file is dependent on the values of `TET_ROOT` and `TET_SUITE_ROOT`, if `TET_ROOT` is set then the file will be placed under it, if `TET_SUITE_ROOT` then it will be placed under there.

`Systems`

This file is only generated if distributed TETware *professional* is used and remote systems have been added to the Test Run. The location of this file is always in the `TET_ROOT` directory.

`tetdist.cfg`

This file is only generated if distributed TETware *professional* is used and remote systems have been added to the Test Run. The location of this file is dependent on the values of `TET_ROOT` and `TET_SUITE_ROOT`, if `TET_ROOT` is set then the file will be placed under it, if `TET_SUITE_ROOT` then it will be placed under there.

Please note: It should be noted that for the same Test Run conditions the files are overwritten when the Test Run is re-executed.

Once all the relevant files have been generated the command line to use to execute TETware *professional* is built. Once the command line has been built TETware *professional* is started with the environment set to the values indicated in the Test Run configuration.

Please note: a 2 second delay has been added to allow time for the Operating System to deliver the captured output to TETware *professional*, it was found during testing that there are times when TETware *professional* can finish execution but the Operating System did not deliver the captured output before TETware *professional* had detected that it had finished. The consequence of this was that TETware *professional* indicated that it had finished but no output was displayed. To circumvent this a delay was added to enable the output from TETware *professional* to be captured correctly.

8.4 Quick Re-Run

When a Test Run has completed if there are some Test Cases that have failed to execute, the failed set of tests may be substantially smaller than the total number of Test Cases in the Test Run. Therefore, it may be desirable to re-execute only part of the Test Run, for example, the failed test cases.

The ability to re-execute quickly only those tests that have a result code of `FAILED` and `UNTESTED` is provided. To access this function the User can:

- Use the `Quick Re-run` menu item in the `Run` menu.
- Right-click on the `Execution Output` tree node, a pop-up menu appears, then click the `Quick Re-run` menu item.

Note: These menu items will only be available when the Test Run is NOT executing.

Whichever way you choose for the `Quick Re-Run`, you will be prompted with a dialog box with the message ‘`Confirm Re-run of Test Run` *`[Name]`*.

If the `Yes` button is clicked then a Quick Re-Run Execution will begin.

If `No` is chosen then the Test Run will be aborted.

Please note that the process of re-executing of a Test Run is the same as for normal execution, this means that all the rules and criteria applicable to normal execution also apply for the `Quick Re-run`. If you have changed any configuration values for the Test Run since it was executed normally then this will have an effect on the values written to the Journal file.

8.5 Advanced Re-Run

There may be times when the User wishes to re-execute a Test Run but wants to have more control over what is re-executed, e.g. they may wish to re-execute all tests that have a result code of `NORESULT`. To do this the 'Advanced Re-run' function is provided, to use this function the User can:

- Use the `Advanced Re-run` menu item in the `Run` menu.
- Right-click on the `Execution Output` tree node and click the `Advanced Re-run` menu item from the pop-up menu.

Note: these menu items will only be available when the Test Run is NOT executing.

Whichever method is chosen a dialog box will prompt you to choose between two options namely

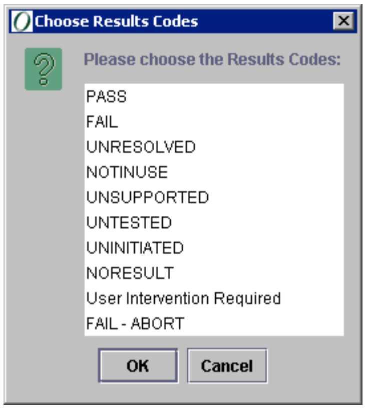
Results Codes: Here the User can specify the results codes that they wish to use for the re-run. This is the default option.

Modes of Operation: Here the User can specify which Modes of Operation they wish to use for the re-run.

8.5.1 Choosing Results Codes

If you choose `Results Codes` and click on `OK` then a dialog box with standard Results Codes together with user defined Results Codes will be displayed, as shown in Figure 24.

You will then be able to choose one or more Results Codes that you wish to use for the Re-run. To select a contiguous list of Results Codes you should hold `'Shift'` and click the desired list of codes. To select a dis-contiguous list of Results Codes you should hold `'Ctrl'` and select the list of codes.



Choose Results Codes

Please choose the Results Codes:

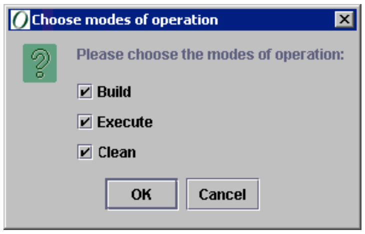
PASS
FAIL
UNRESOLVED
NOTINUSE
UNSUPPORTED
UNTESTED
UNINITIATED
NORESULT
User Intervention Required
FAIL - ABORT

Figure 24: Choose Results Codes

Once you are satisfied with the Results Codes chosen, you should click `OK`, the Test Run will be then be Re-run for the Results Codes specified with the configuration present in the Test Run. Select `Cancel` to abort.

8.5.2 Choosing Modes of Operation

If you choose `Modes of Operation` and click `OK` then the following dialog box will be displayed



Choose modes of operation

Please choose the modes of operation:

☑ Build
☑ Execute
☑ Clean

Figure 25: Choose modes of operation.

This dialog box allows you to specify the Modes of Operation that you wish to use in performing the re-run. You are able to specify one or more Modes of Operation. Note, at least one Mode of Operation MUST be specified to continue.

If one or more Modes of Operation are specified and `OK` is chosen then the Test Run will be Re-run for the modes specified with the configuration present in the Test Run.

If no Mode of Operation is specified and `OK` is chosen then an error dialog box with the message 'Error: No modes of operation chosen, aborting' will be displayed.

8.6 Stop Test Run.

Please note: Stopping a Test Run whilst it is executing is NOT recommended and should be avoided where possible due to the potentially undesirable effects it can have on the system.

To stop a currently executing Test Run the User can:

- Use the `Stop Test Run` menu item in the `Run` menu.
- Right-click the `Execution Output` tree node and click the `Stop Test Run` menu item.

Note: these menu items will only be available if the Test Run is currently executing.

Whichever method is chosen a dialog box will be displayed with the message '`Confirm halt of execution of New Test Run`'. It further asks you if you are sure you wish to continue to stop the Test Run as stopping it may have an un-desirable impact on the system.
If you choose `Yes` then the Test Run will be stopped and the `Execution Output` tree node icon changed to 🖥. However if `No` is clicked, the Test Run will continue.

Note: It may take some time for the process to stop executing, this is due to TETware *professional* trying to perform some clean up before stopping. Please be patient when stopping Test Runs.

8.7 Execution Output

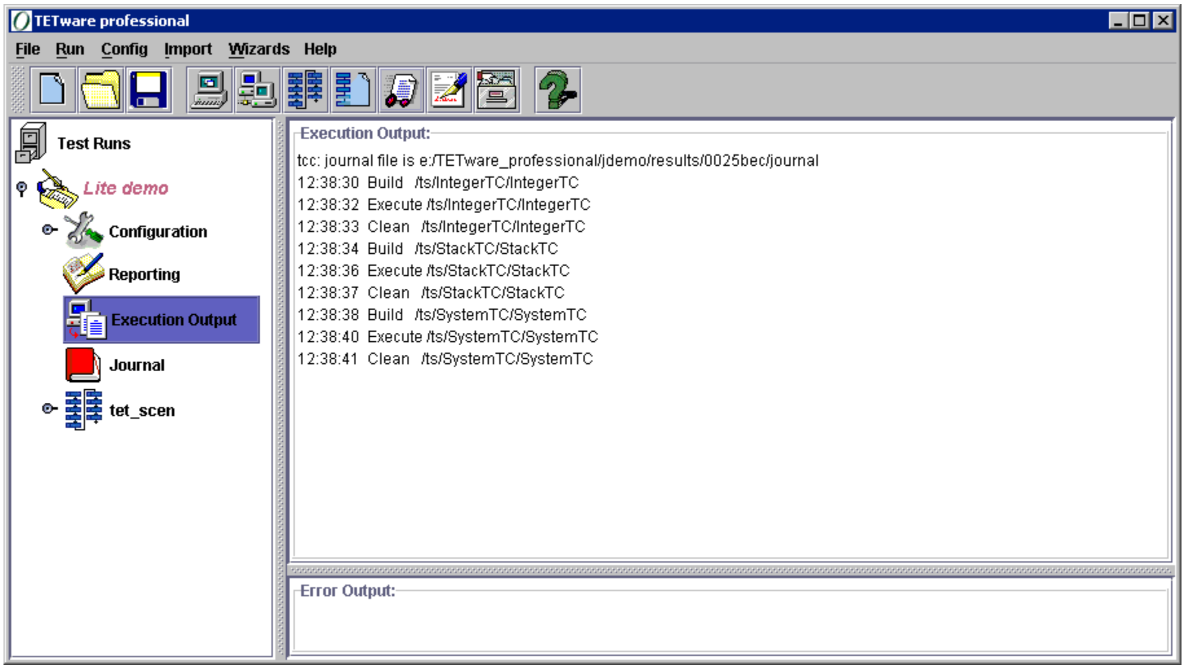
When a Test Run is been running, the icon of Execution Output tree node will change from 🖥 to ⛔; and it will change back to 🖥 once the execution is completed.

Whilst executing a Test Run TETware *professional* will capture the standard output and standard error streams and display them in the "Execution Output" and "Error Output" text areas on the right-hand side of TETware *professional* respectively.

Figure 26 shows a Test Run that has finished executing:

8.7.1 Execution Output

This window will display the path and name of the journal file to which the Output was written. It will also display the time, modes of operation and the Test Cases used during the Test Run.



```
tcc: journal file is e:/TETware_professional/jdemo/results/0025bec/journal
12:38:30  Build   /ts/IntegerTC/IntegerTC
12:38:32  Execute /ts/IntegerTC/IntegerTC
12:38:33  Clean   /ts/IntegerTC/IntegerTC
12:38:34  Build   /ts/StackTC/StackTC
12:38:36  Execute /ts/StackTC/StackTC
12:38:37  Clean   /ts/StackTC/StackTC
12:38:38  Build   /ts/SystemTC/SystemTC
12:38:40  Execute /ts/SystemTC/SystemTC
12:38:41  Clean   /ts/SystemTC/SystemTC
```

Figure 26: Execution Output

8.7.2 Error Output

This window will display any error messages for you after you have executed a Test Run. If you did not get the right Output from your Test Run, you should check in this box for any error messages and make the necessary corrections in TETware *professional*. If you need to contact us for support, you will also need to include the message you have in this field along with your email and any Journal File. Highlight the Error Output text and use `Ctrl+C` to copy it into an email.

8.7.3 Journal File

Once the output has been captured TETware *professional* will determine what the Journal file name is for this Test Run execution. The Journal file name once captured will be used to display the contents of the Journal file in the Journal text area. Please see Section 9 for more details.

8.8 Test Case Locking

When TETware *professional* processes a test case, it prevents possible interference from other instances of tests by acquiring one or more exclusive or shared locks. These locks are acquired in the test case source directory and (optionally) in the test case execution directory. An exclusive lock is acquired by creating a file called `tet_lock` in the directory that is to be locked.

A shared lock is acquired by creating a file with a unique name below a directory called `tet_lock` in the directory that is to be locked, creating the `tet_lock` directory if necessary as well. A diagnostic is printed to the journal file if TETware *professional* is unable to acquire the locks that it needs. Although each instance of this tests is always careful to remove the locks that it created when they are no longer required, it is possible that old locks might remain after some system or process malfunction. In the event of this happening it is necessary to remove the old locks before proceeding (the locks are always located in the directory that contains your test cases).

9 Journal

The journal facility provides a mechanism for outputting data to the execution results file. The API ensures that each entry in this file is written automatically and that there is sequencing information applied to the entry.

9.1 Generating Journal Output

Journal files are automatically generated and stored in sequence under the directory `test_suite_root\results\`*`NNNN`*`[bec]`. (Where *NNNN* represent the sequence number allocated by the program and ‘b e c’ represents the mode of operation: build, execute and clean.

9.2 Viewing a Journal File

If a Test Run has been executed and a Journal file created by TETware *professional* then that Journal file can be viewed by clicking on the name `Journal` on the Tree Node. The Journal file will be parsed and placed into the `Journal Text` area. This behavior is automatic.

10 Reporting

10.1Generating a Report with the TETware Report Writer

If you want to Generate Report, Select the Run `TETware Report Writer` button on the toolbar .

You can then go and view your Report at the specified location (see Section 7.1) using a tool that is appropriate for the format selected.

10.2Database Output

If the Auto-write option is selected on the Database Setup tab (see Section 7.2.3) and all the functions have been set up properly (see Section 7.2) the Journal file parameters will be written to the database as soon as the Journal file is posted by TETware *professional*.

Alternatively you can cause the Journal file parameters to be written to the database by:

1. Clicking of the icon that starts the extraction of the journal file parameters to the Database.

2. The Open window will come up to give you the option of selecting the journal file you wish to extract into the Database. (By default, it will come up with the last run journal file created). Then Click on `Open.` You may however click on the `Cancel` button to abort the operation.

3. If all settings and all parameters are right then the Database message box ‘`Database Extract Finished’` will appear.

4. If an error message appears with the message ‘`Problem with action – There is no Test Run Active`’. Please go back to TETware *professional* main screen and make a Test Run active by clicking on any of the options on the left side of your screen. Then start the process again.

5. If you select an invalid file or path. It will come up with an error message `‘Extract problem` – (The specified path is invalid). You will then need to go back and select a valid path and file.

10.3 Generating your own Report:

It is possible to generate reports for a Test Run using your own Report Writer. See Section 7.3 for Setup instructions.

If the Test Run has been successfully executed then the name of the Journal file used will be pre-filled in the ‘Journal file to use’ text field, you can change this file if they wish and browse the file system for a replacement. Please note: the Journal file MUST exist, if not an error dialog, `Journal file does not exist`, will be displayed.

If all the criteria outlined above are met then press the `Generate` button (see Figure 23) to start the production of a report. When the Report writer has finished and the report generated then the `Report Generation finished` dialog box will be displayed.

You can stop the Report writer at any time by clicking `Stop`.

11 Saving a Test Run

There are four ways to save a Test Run:

- Use the icon 💾 on the toolbar
- Use the ‘`Save`’ or ‘`Save As`’ menu items from the File menu
- Right-click on the `Test Runs tree Node` and click the ‘`Save`’ or ‘`Save` As’ menu items.
- Use the Keyboard Shortcuts Alt-S (Save) or Alt-A (Save As).

11.1 Using the ‘`Save`’ option

You can either use the icon on the toolbars, or the ‘`Save`’ menu item from the File menu, or use the ‘`Save`’ menu by right-clicking on the Test Run tree node, or use the keyboard shortcut Alt-S, you will be prompted with a File selection dialog box.

The file selection dialog box will let you traverse the file system. Once the appropriate file has been chosen and the `Save` button clicked, TETware *professional* will attempt to save the Test Run to the chosen file[4]. If the chosen file already exists, then an Overwrite warning dialog box will be displayed.

If you choose `Yes`, the existing file will be completely overwritten and the Test Run saved to that file instead. If the file does not exist then the chosen file will be created in the file system and the Test Run written to that file.

If you choose `No` the Test Run will not be saved.

Each Test Run that is saved is given an extension of `.trf`, this is for convenience only, and the user is able to change the extension of the Test Run file. A file filter of `*.trf` is provided in the file selection box to allow you to find Test Run files easily and quickly. If the extension is changed, it may then become difficult to find in the future.

[4] Please note: the Test Run is written to the chosen file as a Java serialized object, it is not a text file. Any editing of the Test Run file will probably cause that Test Run to be unusable. The Open Group can accept no responsibility for Test Run files that have been edited by the User. If the ‘Save file’ variables has been set, i.e. by saving the Test Run at any time, then when the ‘Save’ menu item or toolbar icon is clicked then the Test Run will be saved quietly to the chosen save file.

11.2 Using the ‘`Save As`’ Option

Using the ‘`Save As`’ function allows you to save the Test Run as any name you wish at any time, regardless of whether the ‘`Save file`’ variable has been set for a Test Run. When the ‘`Save As`’ function is used, a file selection dialog is displayed. You will be able to browse the file system using the dialog box. Once the selection dialog box is displayed, the procedure is the same as for the ‘`Save`’ functions. (Please see Section 11.1 above).

11.3 Setting Defaults

It is possible to set the values that a Test Run contains as the defaults for that session of TETware *professional*. Once the defaults are set it is then possible to set the configuration values for another Test Run to those values.

It should be noted that setting the defaults for a Test Run is handled in different ways, depending upon whether you try to set the defaults via the "Config" menu or via a right-click on a node in the Test Run.

11.3.1 Config menu default setting

11.3.1.1 Save as default settings

If the "Save as default settings" menu item is chosen from the "Config" menu then TETware *professional* will:

- Set ALL the configuration values in the main configuration (all tabs related to the "Configuration" tree node) as the defaults, this means that any values that are set as blank in the tabs will be saved as blank.

- Set ALL the configuration values in the local system as the defaults, again this means that any values that are set as blank in the tabs will be saved as blank.

It should be noted that if any remote systems have been added to the Test Run then they will NOT have their values saved as the defaults. This is due to the fact that to set the defaults for another Test Run that does not any remote systems would not make any sense in terms of remote systems. If you wish to set the defaults for a remote system please see below.

11.3.1.2 Revert to default settings

If the "Revert to default settings" menu item is chosen from the "Config" menu then TETware *professional* will:

- Set the currently Active Test Runs main configuration to the saved defaults.
- Set the currently Active Test Runs local system configuration to the saved defaults.

It should be noted that if there are any remote systems present in the Test Run then the configuration values associated with that remote system will NOT be set.

11.3.2 Default setting via right-clicking on a tree node

11.3.2.1 Save as default settings

It is also possible to set the defaults for a Test Run via the right-click menus on the various tree nodes. It should be noted that when the defaults are set via the right-click menus then only the configuration associated with that tree node will be set as the defaults. The configuration that you can set the defaults for via the right-click menus are:

- The "Configuration" node, this will set the main configuration values for a Test Run as the defaults.
- The "local system" node, this will set the local systems values for a Test Run as the defaults.
- The "remote system" node, this will set the associated remote systems values as the defaults.

11.3.2.2 Revert to default settings

If the "Revert to default settings" menu item is chosen from a right-hand menu then the configuration values associated with that tree node will be set to the saved defaults.

Please note: If no defaults have not saved and the User chooses to set the defaults (regardless of origin) then the configuration values will be set to blank, there are NO built in default values coded into TETware *professional* defaults.

12. Wizards

The TETware *professional* GUI includes 3 wizards that are included to make the task on configuring Test Run and Remote System easier. They do this by stepping the User through the various components of the configuration tabs in sequence. The wizards are useful because they move between the different tabs to ensure that all the appropriate options are considered. They are largely, self-explanatory.

The three (3) Wizards are detailed in Tabular forms below.

12.1 Test Run Setup (Basic) Wizard

This Wizard steps the User through the four minimum steps that MUST be carried out to configure a new Test Run.

Label	Please See Section
Test Run Name	4.1.1
Test Run Description	4.1.1
Scenario File	5.2
`TET_ROOT`	4.2.1

Table 2: Test Run Setup (Basic) Wizard

12.2 Test Run Setup (Advanced) Wizard

This Wizard steps the User through ALL the steps that MIGHT be carried out to configure a New Test Run. There are 37 steps in this Wizard. The User can click on `Next` to pass through steps that are not relevant to their Test Run setup or Click on `FINISH` to end all setups.

Label	Please See Section
Name	4.1.1
Test Run Description	4.1.2
Scenario File Selection	5.2
`TET_ROOT`	4.2.1
Modes of Operation	4.1.1
Build mode configuration file	4.1.4
Execute mode configuration file	4.1.4
Clean mode configuration file	4.1.4
Include Test Cases	4.1.2
Exclude Test Cases	4.1.2
Custom Result codes	4.1.3
Customize User Configuration Variables	4.1.5
Timeout	4.1.6.1
`TET_XTI_TPI`	4.1.6.2
`TET_XTI_MODE`	4.1.6.2
`TET_LOCALHOST`	4.1.6.2
Journal File Name	4.1.6.3
`TET_SUITE_ROOT`	4.2.1
`TET_EXECUTE`	4.2.1
`TET_TMP_DIR`	4.2.1
`TET_RUN`	4.2.1
`TET_BUILD_TOOL`	4.2.2
`TET_BUILD_FILE`	4.2.2
Build Mode `TET_OUTPUT_CAPTURE`	4.2.2
Build Mode `TET_API_COMPLIANT`	4.2.2
Build Mode `TET_PASS_TC_NAME`	4.2.2
`TET_EXEC_TOOL`	4.2.3
`TET_EXEC_FILE`	4.2.3
Exec Mode `TET_OUTPUT_CAPTURE`	4.2.3
Exec Mode `TET_API_COMPLIANT`	4.2.3
Exec Mode `TET_EXEC_IN_PLACE`	4.2.3
Exec Mode `TET_SAVE_FILES`	4.2.3
`TET_CLEAN_TOOL`	4.2.4
`TET_CLEAN_FILE`	4.2.4
Clean Mode `TET_OUTPUT_CAPTURE`	4.2.4
Clean Mode `TET_API_COMPLIANT`	4.2.4
Clean Mode `TET_PASS_TC_NAME`	4.2.4

Table 3: Test Run Setup (Advanced) Wizard

12.3 New Remote System Wizard

This Wizard steps the User through ALL the steps that MIGHT be carried out to start and configure a `New Remote System`. There is a total of 23 steps in this Wizard. The User can click on `NEXT` to pass through steps that are not relevant to their Remote System setup or Click on `FINISH` to end all setups.

Label	Please See Section
Name	6.1
System Number	6.1
`TET_ROOT`	6.2.1
`TET_SUITE_ROOT`	6.2.1
`TET_EXECUTE`	6.2.1
`TET_TMP_DIR`	6.2.1
`TET_RUN`	6.2.1
`TET_BUILD_TOOL`	6.2.2
`TET_BUILD_FILE`	6.2.2
Build Mode `TET_OUTPUT_CAPTURE`	6.2.2
Build Mode `TET_API_COMPLIANT`	6.2.2
Build Mode `TET_PASS_TC_NAME`	6.2.2
`TET_EXEC_TOOL`	6.2.3
`TET_EXEC_FILE`	6.2.3
Exec Mode `TET_OUTPUT_CAPTURE`	6.2.3
Exec Mode `TET_API_COMPLIANT`	6.2.3
Exec Mode `TET_EXEC_IN_PLACE`	6.2.3
Exec Mode `TET_SAVE_FILES`	6.2.3
`TET_CLEAN_TOOL`	6.2.4
`TET_CLEAN_FILE`	6.2.4
Clean Mode `TET_OUTPUT_CAPTURE`	6.2.4
Clean Mode `TET_API_COMPLIANT`	6.2.4
Clean Mode `TET_PASS_TC_NAME`	6.2.4

Table 4: New Remote System Wizard

13 Help

13.1 TETware *professional* Help Pages (In-Line Help)

All TETware *professional* tabs include help text on each item within the tab.

You can display help information on your screen anytime by clicking on the Help icon, ? or by choosing the Help index command from the Help menu, or by using the Keyboard Shortcut Alt-H.

Online Help for TETware *professional* is set up in such a way that the user will find it easy to move around and collect whatever information or assistant they may require.

13.2 TETworks Web Site (On-line Help)

The Help Pages can also be accessed without opening TETware *professional*. Use your browser to access them at http://tetworks.opengroup.org/Help/Index.htm You will also find them behind the documentation button on the main TETworks site: http://tetworks.opengroup.org

13.3 Creating Your Own Help Pages

Note: The ability to create your own help pages is NOT a supported function within TETware *professional*, it is provided for the Users benefit. The Open Group cannot accept responsibility for any adverse effects it may have on the TETware *professional* installation. No problem reports will be accepted against the TETware *professional* that relate to the production of Personal help pages.

There may be times when it is desirable to have supplementary information relating to the management of Test Runs recorded somewhere that the User may wish to refer to. It is possible to do so with the Help Tree. The Help window can display both plain text and `HTML3.2`.

It is recommended that if you wish to create your own `HTML` based Help Pages then you use the markup features used in the TETware *professional* pre-written help pages.

The Help pages are created as follows:

- At start-up TETware *professional* reads in the `helpstructure.file` that should be present in the `TET_ROOT/help` directory. The `TET_ROOT` directory is the directory that the TETware *professional* was installed to, it will look for the help structure file in the '`help`' directory located off the base directory.

- The `helpstructure.file` is parsed and the help tree built based upon the syntax in the file.

The syntax is outlined below:

Each valid line in the help structure file is composed of 3 parts. Any less than 3 parts and the line will be ignored. The parts are:

Identifier:

The Identifier indicates to the help parser what the type of this line is, the types are described below, and the Identifier should be first thing that is present on the line:

`ROOT:` This indicates the start of the Help Tree structure. It is the starting point of the Tree, do not create more than one `ROOT` since the preceding `ROOT` and its sub tree nodes will be lost.

`H[value]` This is a heading value, with the [value] indicating the depth of the heading. Basically a heading is a folder for the Help Tree. Currently a depth of 1-9 is supported. A depth of 0 will probably cause a trap and the Help Tree not being built and hence displayed. Values greater than 9 will be interpreted as the first value in the number, i.e. 10 will be interpreted as 1, 31 as 3 etc.

`SUB` This is the indicator for a page of data that will have no children, if you create a heading below a `SUB` or list of `SUB`s then it will be added to the appropriate node above the `SUB` or `SUB`s indicated.

Name

This is the name that you would like to be displayed in the Help Tree, it is basically the node name of the item. The name is also displayed in the Window title if you click on the node and there is a page associated with it.

Page

This is the page that you wish to associated with the Tree node, the path to the page should be relative to the `TET_ROOT/help directory`, the path should be a full path specification, i.e. if the page is held in `/TETware_`*`professional`*`/help/MyHelp/my_help_page.html`, then the page specification should be `/MyHelp/my_help_page.html`. The leading `/` is important. If you do not want a page to be associated with a given node then use the text '*null*' instead of a page specification. For folders this will mean that no info indicator is overlaid on the folder icon. For info nodes it doesn't make sense to use 'null' but it is possible, in that case when the icon is clicked no page is displayed.

The Identifier and Name **MUST** be separated by a '`=`', if they are not then the line is considered invalid and ignored.

The Name and Page specification should be separated by a '`|`', again, if they are not then the line is considered invalid and ignored. No blank space should be placed between the '|' and the start of the Page specification as it will be considered part of the specification and switching to that page will not be possible.

The page given for `ROOT` (if present) is then displayed as the start page.

To create your own help pages follow the steps indicated below:

1. **Create your Help Pages** You should create your help pages and place them in a sub directory from the `TET_ROOT/help directory`. You can create any depth of directory structure you need. The help pages can be plain `text` or in a `HTML3.2` format.

   (Note: Currently Java 2 only supports `HTML3.2`, frames are NOT supported).

2. **Create a back-up copy of** `helpstructure.file` Create a back-up copy of `helpstructure.file` to ensure that you are able to undo any changes.

3. **Modify** `helpstructure.file` Modify `helpstructure.file` to include your help pages/folders etc.

4. **Reload TETware *professional*** Once you reload TETware *professional* your help pages should now be visible.

Important Notes

The following are a list of important notes that should be read and followed whenever you create your own Help Pages:

Beware of Images:

Image path specification can be a problem with the `Java HTML` editor. To specify the location of images to be displayed in the Help Page you should use the following format:
<IMG SRC='`file:help`*`/[yourpath]/yourimage.jpg`*'>
The syntax here is very important, the IMG SRC directive MUST start with '`file:help`', no leading / should be placed before '`help`'. GIF images (.gif) are also supported by the Editor.

- **Beware of Image Names:**

  Image names can also be a problem, it has been observed that image names that are all capitals are NOT loaded into the page correctly (this problem has only been observed on Win32 systems). Where possible it is best to use all lower case names for images.

- **Rendering the <CENTER> Tag:**

  It has been observed that help page written in HTML that contains the <CENTER> tag will not be rendered correctly. No reasonable explanation has been found for this, however a work-around using a <BR> tag is effective. Just use:

```
<CENTER>
desired HTML
<BR>
</CENTER>
```

This problem is seen when a node in the help tree is clicked but no associated page is displayed and the nodes color is not changed.

13.3.1Examples

Here are a few examples for creating your own pages:

Creating a New Folder with no Associated Page with Sub-Pages

In general for completely new folders it is best to create a whole new top-level folder, so use:

```
H1=My folder|null
```

Here a new folder is created with no associated page. For sub pages it is best to create whole new directory under the TET_ROOT/help directory, so create:

```
TET_ROOT/help/My_folder/
```

To add sub pages use:

```
SUB=My sub page|/My_folder/my_sub_page.html
SUB=My other sub page|/My_folder/my_other_sub_page.txt
```

If your .html files contains images then create a /images directory from the My_folder directory and place images in there. When referring to the images in the .html file use:

```
<IMG SRC=‘file:help/My_folder/images/my_image.jpg’>
```

Creating Sub Folders

If you wish to create a sub folder from an existing one then you should create a new directory in the file system that branches from the folder you wish to branch from. So if folder `TET_ROOT/help/Getting_started/` already exists and you wish to create a new folder under it then create a directory to hold the files:

```
TET_ROOT/help/Getting_started/My_sub_folder/
```

Then place your help pages in the `My_sub_folder` directory. Modify `helpstructure.file` to add your sub folder, for sub folders you should add one to the number of the heading you wish to branch from, i.e. If the following already exists in `helpstructure.file`,

H1=Getting Started|null

Then, you add something like:

H2=*My starting page*|/Getting_started/My_sub_folder/my_starting_page.html

13.4 Feedback and Problem Reporting

If you have any comments on this manual, TETware *professional* or the TETworks website please email them to tetware_manager@opengroup.org

All Problem Reports are welcome and actively encouraged. The more problems that are found and fixed the better the product will be. Please submit all bugs and queries found. Also, please submit requests for features and upgrades.

Users should submit problems report using the dedicated email address that is provided for them. Evaluators should email tetware_manager@opengroup.org.

14 Exiting TETware *professional*

You can exit TETware *professional* by:

- Using the "Exit" item in the File menu
- Use the system exit

Exiting TETware *professional* is handled differently dependent on the way in which the exit is requested.

Please note: It is recommended that you should always exit TETware *professional* via the "File" menu, this is due to the way in which Frame exiting is handled in Java.

The different methods of exiting TETware *professional* are detailed below:

14.1 Exiting via the File menu

If you exit TETware *professional* via the File menu then the following checks are made:

Whether there are Test Runs currently executing

If there any Test Runs that are currently executing then a dialog box will be displayed to confirm whether you wish to stop execution of the Test Runs. The Stop executing test runs dialog box is displayed.

Once you click `OK` all executing Test Runs will be stopped. This may take some time since TETware *professional* will perform a limited clean up of the Test Runs execution.

If any Test Runs are present in TETware *professional*

If there any Test Runs open in TETware *professional* you will be prompted to save the Test Runs, saving is not mandatory but is advisable. The Confirm save test runs dialog box is displayed.

If you choose `OK` then all the Test Runs currently open in TETware *professional* will be saved. If the Test Runs have save files associated with them then they will be saved quietly to those files, if not then the `Save As` dialog box will be displayed. For more information on saving Test Runs please see Section 11. If you choose `No` the Test Runs will not be saved and TETware *professional* will exit. The Java Interpreter will also exit.

Once the Test Runs have been stopped (if any are executing) and they have been saved, you will be prompted to exit TETware *professional* with the Confirm exit from TETware *professional* dialog box. If you choose `Yes` in this dialog box then TETware *professional* will exit and the Java Interpreter will also exit. If you choose `No` then TETware *professional* will NOT exit.

14.2 Using the system exit

This method of exiting TETware *professional* is dangerous and not recommended if there any Test Runs currently executing. There is no way to halt TETware *professional* closing once a system exit has been called. This means that TETware *professional* is not able to cancel the exit and must try and clean up any Test Runs before exit. TETware *professional* will perform the following checks on exit:

Whether there are Test Runs currently executing

If there any Test Runs that are currently executing then a dialog box will be displayed indicating that any currently executing Test Runs will be stopped. The Stop test run execution dialog box is displayed.

Once you click `OK` all executing Test Runs will be stopped. This may take some time since TETware *professional* will perform a limited clean up of the Test Runs execution.

If any Test Runs are present in TETware *professional*

If there any Test Runs open in TETware *professional* you will be prompted to save the Test Runs, saving is not mandatory but is advisable. The Confirm save test runs dialog box is displayed.

If you choose `OK` then all the Test Runs currently open in TETware *professional* will be saved. If the Test Runs have save files associated with them then they will be saved quietly to those files, if not then the `Save As` dialog box will be displayed. For more information on saving Test Runs please see Section 11. If you choose either `No` then no Test Runs will be saved and TETware *professional* will exit. The Java Interpreter will also exit.

Appendix A - TETware *professional* License

+++++++++++++TET END USER LICENSE+++++++++++

BY DOWNLOADING THIS PRODUCT, YOU ARE CONSENTING TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT AGREE TO ALL OF THE TERMS OF THIS AGREEMENT, DO NOT INSTALL THE PRODUCT.

TETWARE PROFESSIONAL RELEASE 1 END USER LICENSE
REDISTRIBUTION NOT PERMITTED

This Agreement has two parts, applicable to the distributions as follows:

(A) Free binary evaluation copies - valid for up to 45 days, full functionality - no warranty,

(B) Licensed versions - full functionality, warranty fitness as described in documentation, includes annual support.

PART I (A above) -- TERMS APPLICABLE WHEN LICENSE FEES NOT (YET) PAID (LIMITED TO EVALUATION USE)

GRANT.

X/Open Company Limited, trading as The Open Group ('The Open Group') grants you a non-exclusive license to use the Software free of charge if your use of the Software is for the purpose of evaluating whether to purchase an ongoing license to the Software. The evaluation period for use by or on behalf of any entity is limited to a maximum of 90 days. If you are using the Software free of charge you are not entitled to hard-copy documentation, support or telephone assistance. If you fit within the description above, you may use the Software for any purpose and without fee.

DISCLAIMER OF WARRANTY.

Free of charge Software is provided on an 'AS IS' basis, without warranty of any kind.

THE OPEN GROUP DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT

SHALL THE OPEN GROUP BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

PART II (B above) -- TERMS APPLICABLE WHEN LICENSE FEES PAID

GRANT. Subject to payment of applicable license fees, The Open Group grants to you a non-exclusive license to use the Software and accompanying documentation (‘Documentation’) as described below.

LIMITED WARRANTY.

The Open Group warrants that for a period of ninety (90) days from the date of acquisition, the Software, if operated as directed, will substantially achieve the functionality described in the Documentation. The Open Group does not warrant, however, that your use of the Software will be uninterrupted or that the operation of the Software will be error-free or secure.

EXCEPT AS STATED ABOVE THE OPEN GROUP DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE OPEN GROUP BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

SCOPE OF GRANT.

Permission to use for any purpose is hereby granted, as long as a support contract is in place.

Modification of the source is permitted.

Redistribution of the source code is not permitted without express written permission of The Open Group. Distribution of sources containing adaptations is expressly prohibited.

Redistribution of binaries or binary products containing TETware code is not permitted unless the distributor has a redistribution agreement with The Open Group.

Modifications sent to the authors are humbly accepted and it is their prerogative to make the modifications official.

Portions of this work contain code derived from other versions of the Test Environment Toolkit, which are copyright

Copyright 1990,1992 Open Software Foundation
Copyright 1990,1992 Unix International
Copyright 1990,1992 X/Open Company Ltd.
Copyright 1991 Hewlett-Packard Co.
Copyright 1993 Information-Technology Promotion Agency, Japan
Copyright 1993 Sunsoft, Inc.
Copyright 1993 UNIX System Laboratories, Inc., a subsidiary of Novell Inc.
Copyright 1994,1995 UniSoft Ltd.

The unmodified source code of those works is freely available from ftp.xopen.org. The modified code contained in TETware *professional* restricts the usage of that code as per this license.

+++

Appendix B - TETware *professional* Database Design

Why did we Create the 'Database Extract Tool'?

Our objective is to provide TETware *professional* users with the ability to store test results in a Standard Database from which they can extract queries and reports providing detailed comparative information relating to test suites, Test Runs and individual test cases.

The extract tool is designed to be compatible with any SQL compliant Database. It extracts journal file parameters and passes them into a Database. Users may then use either the queries or reports provided, or define their own queries and reports using their preferred tools.

`TETware Report Writer` facility provides users with a report on a particular Test Run. The Database facility offers the added advantage of allowing users to compare the results from several executions of the same test or Test Run. As such, it provides a powerful test management tool.

It also allows users to add their own database tables so that they may generate reports containing information that is not provided by TETware *professional*.

What are its Features and Abilities?

The Database extract tool provides users with the means of easily transferring TETware *professional* journal file parameters into most Databases.

It uses standards in order to achieve this:

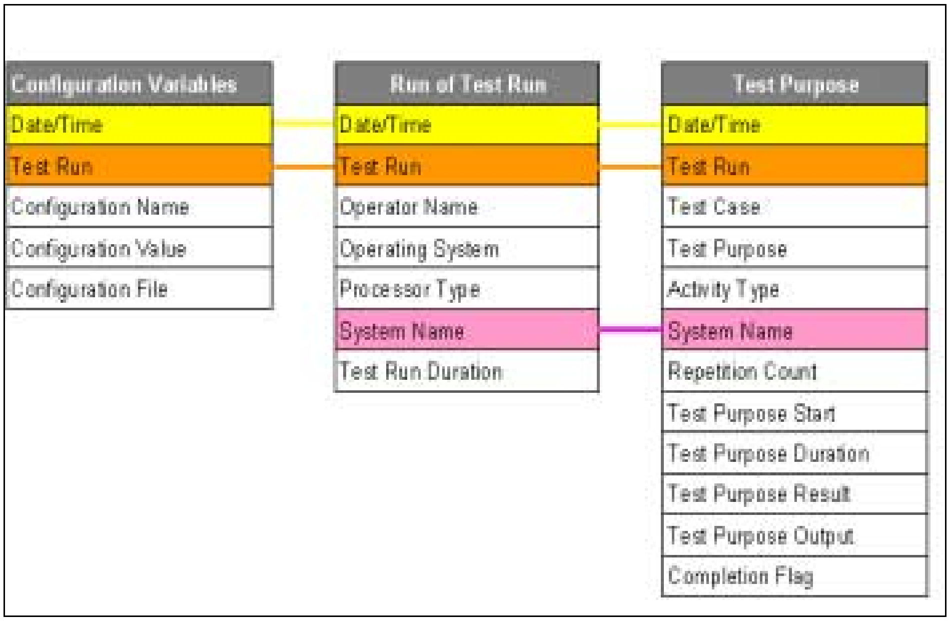
- Standard Structured Query Language (SQL);
- Open Database Connectivity (ODBC);
- Java® Database Connectivity (JDBC).

It is a fully supported component of TETware *professional,* so users who encounter difficulties in making their Database connection can use the email support facility to request help from The Open Group Support Team. If you are evaluating TETware *professional* please email your questions to tetware_manager@opengroup.org.

Database Tables

TETware *professional* Database comprises three tables. The relationship between the Database tables is shown in Table 6.

- **Configuration Variables**: This table lists the settings used on Test Runs to specify information about that Test Run. *e.g.* Configuration Name, Value and File.
- **Run of Test Run**: This table lists the header information that the user requested. *e.g.* Operators Name, Operating System, Processor type.
- **Test Purpose**: This table lists the detail of the Test Run. This may include information such as the test result, test duration, the time that the test started.



Configuration Variables
Date/Time
Test Run
Configuration Name
Configuration Value
Configuration File

Run of Test Run
Date/Time
Test Run
Operator Name
Operating System
Processor Type
System Name
Test Run Duration

Test Purpose
Date/Time
Test Run
Test Case
Test Purpose
Activity Type
System Name
Repetition Count
Test Purpose Start
Test Purpose Duration
Test Purpose Result
Test Purpose Output
Completion Flag

Table 5: Relationship between Database Tables

Database Fields

Within each table, there are a number of fields. In the specimen Databases provide these fields have properties which are defined in the tables below:

Run of Test Run

Field Name	Data Type	Description	Field Size	Format	Decimal Places	Required	Indexed
Date/Time	Date/Time	Defines particular run of a Test Run	n/a	General Date		Yes	Yes (No Duplicates)
Command Line	Text	Command line passed to controller	255	Normal		No	No
Test Run	Text	Test Run Name	255	Normal		No	Yes (No Duplicates)
Operator Name	Text	From First Line 'User:'	255	Normal		No	No
Operating System	Text	From System Information Line	255	Normal		No	No
Processor Type	Text	From System Information Line	255	Normal		No	No
System Name	Text	From System Information Line	255	Normal		No	No
Test Run Duration	Number	From Test Case Controller Start and End lines	Long Integer	General Number	0	No	No

Table 6: Run of Test Run Table

Configuration Variables

Field Name	Data Type	Description	Field Size	Format	Decimal Places	Required	Indexed
Date/Time	Date/Time	Defines particular run of a Test Run	n/a	General Date		Yes	Yes (No Duplicates)
Test Run	Text	Test Run Name	255	Normal		No	Yes (No Duplicates)
Configuration Variable Name	Text	Configuration Name	255	Normal		No	No
Configuration Variable Value	Text	Configuration Value	255	Normal		No	No
Configuration File Name	Text	Configuration File	255	Normal		No	No

Table 7: Configuration Variables Table

Test Purpose

Field Name	Data Type	Description	Field Size	Format	Decimal Places	Required	Indexed
Date/Time	Date/Time	Defines particular run of a Test Run	n/a	General Date		Yes	Yes (No Duplicates)
Test Run	Text	Test Run Name	255	Normal		No	Yes (No Duplicates)
Test Case	Text	Test case name	255	Normal		No	Yes (Duplicates OK)
Test Purpose	Number	Test purpose number	Integer	General Number	0	No	Yes (Duplicates OK)
Activity Type	Text	Build/Clean/Exec	5	Normal		No	Yes (Duplicates OK)
System Name	Text	Name of system participating in the test	255	Normal		No	No
Repetition Count	Number	Sequence number for repeat directives	Integer	General Number	0	No	No
Test Purpose Start	Date/Time	Start time for test purpose	n/a	Long Time		No	No
Test Purpose Duration	Number	Duration of test purpose	Integer	General Number	0	No	No
Test Purpose Result	Text	Result of test purpose	10	Normal		No	No
Test Purpose Output	Text	Output from test purpose	255	Normal		No	No
Test Purpose Output Line	Number	Output Line	Long Integer	General Number	0	Yes	Yes (Duplicates OK)
Completion Flag	Text	Flag indicating tests that failed to complete	1	Normal		No	No

Table 8: Test Purpose Table