TETware

Release Notes
for TETware Release 3.7
TET3-RN-3.7

Released: May 2003

THE *Open* GROUP

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

This document is produced by The Open Group at:

Apex Plaza
Forbury Road
READING
Berkshire
RG1 1AX
United Kingdom

+++++++++++++++++++++++++++ TET END USER LICENCE +++++++++++++++++++++++++++

BY OPENING THE PACKAGE, 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 AND RETURN IT TO THE PLACE OF PURCHASE FOR A FULL REFUND.

TETWARE RELEASE 3.7 END USER LICENCE
REDISTRIBUTION NOT PERMITTED

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

A. Free binary evaluation copies − valid for 90 days, full functionality − no warranty.

B. Free binary restricted versions − no warranty, limited functionality.

C. Licenced versions − full functionality, warranty fitness as described in documentation, includes source, binary and annual support.

PART I (A & B above) − TERMS APPLICABLE WHEN LICENCE FEES NOT (YET) PAID (LIMITED TO EVALUATION, EDUCATIONAL AND NON-PROFIT USE).

GRANT.

X/Open grants you a non-exclusive licence to use the Software free of charge if

a. you are a student, faculty member or staff member of an educational institution (K-12, junior college, college or library) or an employee of an organisation which meets X/Open's criteria for a charitable non-profit organisation; or

b. your use of the Software is for the purpose of evaluating whether to purchase an ongoing licence to the Software.

The evaluation period for use by or on behalf of a commercial entity is limited to 90 days; evaluation use by others is not subject to this 90 day limit. Government agencies (other than public libraries) are not considered educational or charitable non-profit organisations for purposes of this Agreement. 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.

X/OPEN DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL X/OPEN 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 (C above) − TERMS APPLICABLE WHEN LICENCE FEES PAID.

GRANT.

Subject to payment of applicable licence fees, X/Open grants to you a non-exclusive licence to use the Software and accompanying documentation (''Documentation'') as described below.

++

1. Introduction

1.1 Preface

These release notes accompany TETware Release 3.7.

TETware includes all of the functionality of the Test Environment Toolkit Release 1.10 (TET), the Distributed Test Environment Toolkit Version 2 Release 2.3 (dTET2) and the Extended Test Environment Toolkit Release 1.10.3 (ETET), together with a number of new features.

There are two principle versions of TETware. One version is known as Distributed TETware. This version uses a client-server architecture and provides support for processing local, remote and distributed test cases. The other version is known as TETware-Lite. This version does not use a client-server architecture or require a network transport. It is more simple to set up than is Distributed TETware but does not provide support for the processing of remote or distributed test cases.

Distributed TETware is implemented on UNIX operating systems and also on the Windows NT operating system. TETware-Lite is implemented on UNIX operating systems and also on the Windows NT and Windows 9*x* operating systems.

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

1.2 Audience

This document is intended to be read by software engineers and/or systems administrators who will install TETware on their computer systems. A knowledge of system administration is assumed when TETware installation and configuration instructions are presented. In addition, a knowledge of network administration is assumed when TETware is to be built to use network transports.

1.3 Conventions used in this document

The following typographic conventions are used throughout this document:

- `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:

- *Test Environment Toolkit: TETware Installation Guide*
  There is one version of this document for each operating system family on which TETware is implemented.

- *Test Environment Toolkit: TETware User Guide*

- *Test Environment Toolkit: TETware Programmers Guide*

- *Test Environment Toolkit: TETware Knowledge Base*

Source and postscript versions of these documents are included in the TETware distribution.

1.5 Problem reporting

If you have subscribed to TETware support and you encounter problems when building or executing TETware you should take a copy of the error reporting template contained in the file *tet-root*/doc/tet3/err.template in the distribution, fill in the details of the problem, and send it by electronic mail to:

        tet_support@opengroup.org

Alternately you may use a web browser to complete and submit the HTML form included in the file *tet-root*/doc/tet3/errtemp.html in the distribution.

You should include sufficient information in your report to enable someone who is unfamiliar with your system to be of assistance in solving the problem.

Users are reminded that the user-contributed software which accompanies the TETware distribution is not covered by TETware support services.

2. New features in this release

The following features appear for the first time in this release of TETware:

- The C TCM/API is enhanced to enable realtime signals to be left alone or ignored, both at compile time and at runtime. The ability to compile the C TCM/API to leave realtime signals alone is useful when building the Java API runtime support library on certain platforms; notably Linux.

  The compile-time functionality is mentioned in the section entitled ''Support for Java'' in Chapter 3 of the TETware Installation Guide for UNIX Operating systems. The runtime functionality is described in the section entitled ''Insulating from the test environment'' in Chapter 8 of the TETware Programmers Guide.

- The Perl API is upgraded to use perl 5 syntax such that it executes without warnings when run under the control of perl −w (but see ''API status'' later in this document).

- tcc correctly identifies Windows 98 and Windows Me when the System Information line is written to the journal.

- In Distributed TETware, an optional Transfer Source Files phase is added at the start of the build mode processing of each test case. In this operation, tcc reads file and directory names from a Source File Transfer instruction file on the local system and copies the corresponding files from the test case source directory on the local system to the test case source directory on one or more remote systems. In simple cases a single instruction file may be provided for use with every test case in a test suite. In more complex cases a separate instruction file may be provided for use with each test case in a test suite. Both ASCII and binary file transfers may be specified.

  This functionality is described in the section entitled ''Transferring source files to remote systems'' in Chapter 3, and the description of TET_TRANSFER_SOURCE_FILES in Chapter 5, both in the TETware Programmers Guide. The format of the Source File

Transfer instruction file is described in the section entitled ''Source file transfer instructions'' in Chapter 6 of the TETware Programmers Guide.

- In Distributed TETware, a test suite may provide a file that associates a file type (that is: ASCII or binary) with a file name suffix. When `TET_TRANSFER_SAVE_FILES` is true, these associations are used to determine the type of copy operation that should take place during the Save Files phase of execute mode processing. These associations are also used in the Transfer Source Files phase of build mode processing when no explicit file type is specified in the Source File Transfer instruction file.

  The format of the File Type Specifications file is described in the section entitled ''File type specifications'' in Chapter 6 of the TETware Programmers Guide.

- Source code for the generic report writer is included in the TETware source distribution.

3. Status of this release

This release of TETware is a general release for production use.

4. Problems fixed since the last release

The following problems have been fixed since the last TETware release:

- In Distributed TETware, there was a problem relating to the way in which the common client/server i/o code handled the receipt of multiple `SIGPIPE` signals.

- Signal handling did not work in the Perl TCM/API.

- When rerun or resume mode resulted in `tcc` processing a scenario directive that had no selected test cases within its scope, `tcc` could go into an infinite loop under certain circumstances.

- On a UNIX system, when `TET_EXEC_IN_PLACE=false` and a test case created a symbolic link below the temporary directory which pointed to a non-existent file, this could result in `tcc` being unable to remove the temporary directory.

- On a UNIX system, when `TET_EXEC_IN_PLACE=false` and a test case created a non-writable or non-searchable directory below the temporary directory, this could result in `tcc` being unable to remove the temporary directory.

5. Known problems in this release

There are no known problems in this release.

6. Building and installing TETware

6.1 Building and installation instructions

For information on how to build and install TETware, please following the instructions in the version of the TETware Installation Guide which is appropriate for your system.

Note: In this release the Java API may be built on Solaris, Linux and Win32 systems. In order to build the Java API on these systems a variable must be set in the `defines.mk` file

which specifies where the Java Development Kit (JDK) has been installed on your machine.

On Solaris Release 7 and later the JDK is supplied with the operating system and so is installed in a standard place (`/usr/java`). This location is specified in the `defines.mk` files for such systems that are supplied in the distribution.

However, on other platforms the JDK might be installed anywhere, so it is necessary to customise your `defines.mk` file if you want to build the Java API. Refer to the section entitled ''Support for Java'' in the TETware Installation Guide for instructions on how to do this.

In the `defines.mk` file on UNIX systems it is necessary to specify the list of signals that are used by the Java Virtual Machine, so as to avoid conflict with the use of signals by the TETware Java API runtime support library. This signal list may change, depending on which JDK and/or operating system version you are using.

The `defines.mk` files for various UNIX platforms on which the Java API is supported contain signal lists for particular JDK versions. If you are using a different JDK version you may need to change this list. Please refer to the section entitled ''Support for Java'' in the TETware Installation Guide for UNIX Operating Systems for further details.

6.2 Installed platforms

6.2.1 UNIX systems

Both TETware-Lite and Distributed TETware versions have been installed and tested on the following platforms:

- AIX Release 4.3.1 using `c89`
- HP-UX Release 11.00
- Linux version 2.2.16 (Slackware) using `gcc` version egcs-2.91.66 and `glibc`
- Solaris Release 9 using `gcc` version 3.2.2
- UNIX System V Release 4.2

6.2.2 Win32 systems

TETware-Lite and Distributed TETware have been installed and tested on Intel PCs running Windows NT release 4.0.

TETware-Lite has been installed and tested on Intel PCs running Windows 98.

6.3 API status

6.3.1 Thread-safe APIs

The Thread-safe APIs have been exercised on the following platforms:

- Linux using POSIX threads
- Solaris using POSIX threads and UI threads

- UNIX System V Release 4.2 using UI threads

- Win32 systems

The following behaviour has been observed when using the Thread-safe API on UnixWare:

i.   A call to `tet_printf()` in a non-main thread causes the process to receive a `SIGSEGV` signal.

ii. Cleanup of left-over threads sometimes causes a fatal error.

It is believed that this behaviour may be due to a problem with the threads implementation on this platform.

On Win32 systems the thread-safe APIs must be used with the multi-threaded DLL version of the C runtime support library. Use with the multi-threaded static version of the C runtime support library is not supported.

6.3.2 C++

The C++ APIs have been exercised on Linux, UnixWare and Win32 systems.

6.3.3 Java

The Java API is supported on Linux, Solaris version 2.7 and later, and Win32 systems.

The Java API has been exercised on the following platforms:

- JDK v1.2 on Linux 2.2.16 using native threads

- JDK v1.4 Solaris Release 9 using native threads

- JDK v1.1 on Windows NT 4.0

6.3.4 Perl

As indicated previously, the Perl API is upgraded to use perl 5 syntax such that it executes without warnings when run under the control of `perl -w`. However, test case authors should note that perl test cases may still emit warnings, when run under the control of `perl -w`, about API interface variables defined in the test case only being used once. For example, consider the following trivial perl test case:

```
#!/usr/bin/perl

@iclist=("ic1");
@ic1=("tp1");

$tet'startup = "startup";
$tet'cleanup = "cleanup";

sub startup
{
        &tet'infoline("in startup function");
}

sub cleanup
{
        &tet'infoline("in cleanup function");
}
```

The Open Group

```
sub tp1
{
        &tet'infoline("This is tp1 in a simple perl test case");
        &tet'result("PASS");
}

require "$ENV{\"TET_ROOT\"}/lib/perl/tcm.pl";
```

When run with `perl −w`, the following warnings are generated from the test case source file:

```
Name "tet::startup" used only once: possible typo at simple.pl line 6.
Name "tet::cleanup" used only once: possible typo at simple.pl line 7.
Name "main::ic1" used only once: possible typo at simple.pl line 4.
Name "main::iclist" used only once: possible typo at simple.pl line 3.
```

When Perl version 5.6 or later is used, it is possible to suppress these warnings by use of the `our` keyword in the test case source file. For example:

```
#!/usr/bin/perl

@iclist=("ic1");
@ic1=("tp1");

$tet'startup = "startup";
$tet'cleanup = "cleanup";

our(@iclist, @ic1);                                                   ←

{                                                                     ←
        package tet;                                                  ←
        our($startup, $cleanup);                                      ←
}                                                                     ←
```
rest of test case …

In this example the code that has been added in order to suppress warning messages is indicated by a ← character in the right margin.

6.4 Transport-specific status

6.4.1 Socket network interface

Versions of Distributed TETware using the socket network interface have been tested on all the platforms listed previously.

6.4.2 XTI network interface

Versions of Distributed TETware using the XTI network interface have been tested on AIX 4.3, Solaris Release 7 and UnixWare 2.1.1 using TCP as the underlying transport provider.

There is a known problem with some SVR4 XTI implementations in which the `t_sync()` function does not work correctly. An XTI implementation may hold transport endpoint data either in kernel or in user address space. The `t_sync()` function is included in the XTI specification for the benefit of implementations which do not automatically detect when the transport endpoint data held in user space is lost. Examples of when such loss might occur are when the file descriptor underlying a transport endpoint is duplicated using `fcntl()` or when the process address space is overlaid by one of the `exec()` system calls.

Since the XTI version of TETware may perform both of these operations, it will not function when used with an XTI implementation which does not automatically detect the loss of data held in user address space and in which `t_sync()` does not perform the advertised function. A common symptom of this problem is when one of the servers `tetsyncd` and `tetxresd` fails with a TBADQLEN error associated with a `t_listen()` call soon after being started by `tcc`.

The XTI version of Distributed TETware has not been tested using an OSI connection orientated transport provider.

7. TETware for TET and ETET users

Some hints and tips for users experienced with previous TET implementations are presented in an appendix to the TETware User Guide.

The Open Group