Compaq DECset for OpenVMS

Compaq Digital Test Manager for OpenVMS Version 4.0 Release Notes

August 2000

These release notes for the Compaq Digital Test Manager for OpenVMS Version 4.0 software contain important general information, new and changed features, current restrictions, and documentation notes.

Operating System and Version:	Compaq OpenVMS VAX Operating System, Versions 6.2, 7.1, 7.2 Compaq OpenVMS Alpha Operating System, Versions 6.2, 7.1, 7.2-1 Compaq DECwindows Motif for OpenVMS, Versions 1.2-4, 1.2-5
Software Version:	Compaq DECset for OpenVMS, Version 12.4 Compaq Digital Test Manager for OpenVMS, Version 4.0

Compaq Computer Corporation
Houston, Texas

© 2000 Compaq Computer Corporation
© Electronic Data Systems Corporation 2000

COMPAQ, VAX, VMS, the Compaq logo, and the DIGITAL logo Registered in U.S. Patent and Trademark Office.

DEC Ada, DECforms, DECnet, DECset, DECwindows, eXcursion, OpenVMS, and PATHWORKS are trademarks of Compaq Information Technologies Group, L.P.

Microsoft, Visual C++, Windows, Windows 95, Windows 98, Windows 2000, and Windows NT are trademarks of Microsoft Corporation.

Motif is a registered trademark of The Open Group.

Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc.

All other product names mentioned herein may be the trademarks or registered trademarks of their respective companies.

Confidential computer software. Valid license from Compaq, EDS, or authorized sublicensor required for possession, use, or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Neither Compaq nor EDS shall not be liable for technical or editorial errors or omissions contained herein.

The information in this publication is subject to change without notice and is provided "AS IS" WITHOUT WARRANTY OF ANY KIND. THE ENTIRE RISK ARISING OUT OF THE USE OF THIS INFORMATION REMAINS WITH RECIPIENT. IN NO EVENT SHALL COMPAQ OR EDS BE LIABLE FOR ANY DIRECT, CONSEQUENTIAL, INCIDENTAL, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, OR LOSS OF BUSINESS INFORMATION), EVEN IF COMPAQ OR EDS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE FOREGOING SHALL APPLY REGARDLESS OF THE NEGLIGENCE OR OTHER FAULT OF EITHER PARTY AND REGARDLESS OF WHETHER SUCH LIABILITY SOUNDS IN CONTRACT, NEGLIGENCE, TORT, OR ANY OTHER THEORY OF LEGAL LIABILITY, AND NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY.

The limited warranties for Compaq and EDS products are exclusively set forth in the documentation accompanying such products. Nothing herein should be construed as constituting a further or additional warranty.

This document was prepared using VAX DOCUMENT, Version 2.1.

Contents

Preface v

1 General Information

1.1 Introduction 1–1
1.2 Associated Documents 1–1
1.3 Installation Requirements 1–2

2 New Features

2.1 Single Digit Time Filter 2–1
2.2 Prevent Purge of Result Files 2–1
2.3 Test Abandonment Following Prologue Error 2–2
2.4 User Defined Filters 2–2
2.4.1 New FILTER Command Qualifier 2–3
2.4.2 New Variables for User Filters 2–4
2.4.3 Filtering Benchmarks on Recording Interactive Terminal Tests 2–4
2.4.4 Using the Global Replace Procedure 2–5
2.4.5 Examples of User Filters 2–6

3 Problems Fixed

4 Known Problems

4.1 General 4–1
4.2 DECwindows Interface 4–3
4.3 Documentation 4–5

5 Restrictions

6 Reporting Problems and Suggestions

Preface

These release notes characterize the Version 4.0 release of the *Compaq Digital Test Manager for OpenVMS (DTM)* software. These notes contain information that either supplements or supersedes the information in the Compaq DECset for OpenVMS documentation set.

Intended Audience

These release notes are intended for users and system managers who need information about DTM on either of the Compaq OpenVMS Operating System platforms (VAX or Alpha).

Document Structure

These release notes are organized as follows.

- Chapter 1 contains general information about this release.
- Chapter 2 contains a description of the new features.
- Chapter 3 describes all problems resolved by the current release.
- Chapter 4 describes all known problems in the current release.
- Chapter 5 describes all restrictions.
- Chapter 6 describes how to report problems.

Sending Compaq Your Comments

We welcome your comments on these release notes or on any other pieces of documentation. See Chapter 6 for details on how to contact Compaq with your comments.

Conventions

The following conventions are used in these notes.

Convention	Description
$	A dollar sign ($) represents the OpenVMS DCL system prompt.
Ctrl/*x*	The key combination Ctrl/*x* indicates that you must hold down the key labeled Ctrl while you press another key, for example, Ctrl/Y or Ctrl/Z or a pointing device button.
boldface text	Boldface text represents the introduction of a new term.
`monospace boldface text`	Boldface monospace text represents user input in interactive examples.
italic text	Italic text represents book titles, parameters, arguments, and information that can vary in system messages (for example, Internal error *number*).
UPPERCASE	Uppercase indicates the name of a command, routine, the name of a file, the name of a file protection code, or the abbreviation of a system privilege.
lowercase	Lowercase in examples indicates that you are to substitute a word or value of your choice.

1
General Information

The information in these release notes pertain to the Compaq Digital Test Manager for OpenVMS (DTM), Version 4.0 software. Compaq suggests that you upgrade your environment incrementally with each new release but there is no technical requirement to do so. It is recommended that you review the release notes for each version because they capture important release-oriented information and advise you of changes between versions.

1.1 Introduction

DTM Version 4.0 is a minor release.

1.2 Associated Documents

The following documents contain information related to DTM. Note however, that for the Compaq DECset for OpenVMS Version 12.4 release, only the software product descriptions, installation guide, and component product release notes have been updated.

The documents associated with DTM are:

- *Guide to DIGITAL Test Manager for OpenVMS Systems*
- *DIGITAL Test Manager for OpenVMS Reference Manual*

Also applicable are the general DECset documents:

- *Compaq DECset for OpenVMS Installation Guide*
- *Compaq DECset for OpenVMS VAX Software Product Description*
- *Compaq DECset for OpenVMS Alpha Software Product Description*
- *Guide to Detailed Program Design for OpenVMS Systems*
- *Using DECset for OpenVMS Systems*

In addition to the base DTM documentation, the DECset engineering team has also provided the *Compaq DECset for OpenVMS Cookbook for an Integrated Product Development Environment*. This document comprises a series of technical articles that describe how each of the component applications can be used in different programming environments. This document is available in PostScript (.PS), HTML (.HTML), and ASCII text (.TXT) formats from the following location:

```
SYS$SYSROOT:[SYSHLP.EXAMPLES.DECSET]DECSET_COOKBOOK.*
```

1.3 Installation Requirements

For a new installation of DTM the following is required:

OpenVMS Alpha Systems

Global Pages	3000
Global Sections	10
Disk Blocks	25000

OpenVMS VAX Systems

Global Pages	800
Global Sections	15
Disk Blocks	15000

2
New Features

The following enhancements were added to DTM Version 4.0.

2.1 Single Digit Time Filter

A new logical variable DTM$TIME_FILTER_NO_SPACE has been added to provide control over the TIME filter.

If this logical is not defined or the value is zero, a space preceding a single digit hour will be absorbed by the filter (the Version 3.9 behavior). If the logical is defined with the value "1", a space preceding a single digit hour will not be absorbed.

This logical can be used in situations where no space is added before a time value when the hour is a single digit. It allows the full range of hours to be filtered to the same string. For example, the following strings will be filtered differently by default, but in the same way if DTM$TIME_FILTER_NO_SPACE is defined as "1":

```
Transaction complete at 4.34
Transaction complete at 11.47
```

2.2 Prevent Purge of Result Files

A new logical variable DTM$RETAIN_RESULTS has been added. When this logical is not defined or is set to a value of zero, the result file will be purged as in Version 3.9. If the logical is defined and set to a value of "1", the result file will not be purged.

The option to retain the original result may be useful in cases where the reason for a failure is obscured by filtering of the result file.

Note that all result files will still be deleted if the test is successful.

2.3 Test Abandonment Following Prologue Error

The procedure which executes each test has been modified to test the status returned by the test prologue. If the status is Error or Severe Error, the test template will not be executed. The test status will then be Not Run.

In order to provide backward compatibility, the new logical variable DTM$IGNORE_PROLOGUE_ERRORS has been introduced. If this logical is not defined or has the value zero, an error status returned by the test prologue will cause the test template to be skipped. If this logical has the value "1", errors in the test prologue will be ignored, retaining the Version 3.9 behavior.

Note that Warning and Informational states will not prevent the test running. Also, collection prologue errors and test and collection epilogue errors will continue to be ignored.

2.4 User Defined Filters

The user defined filter facility provides a mechanism whereby filters written as DEC Text Processing Utility (DECTPU) programs are automatically executed when tests are run. These filters are referred to as user filters. The implementation enables users to solve easily many filtering problems, often with a single-line program, while allowing full access to the facilities of DECTPU to solve more complex cases.

To implement a new filter, a file containing the required DECTPU commands is created. There are a number of predefined patterns and a global replace procedure provided which can be used to build the commands. For example, the following command will replace device names that precede a directory with the string "DEVICE":

```
global_replace( identifier + ':[' , 'DEVICE:[' )
```

The filter can be tested using the new /USER_FILTER qualifier to the FILTER command. For example:

```
FILTER/USER_FILTER=DEVICE.TPU TEST.DAT
```

Alternatively, filters can be developed using the TPU pattern style feature of Compaq Language-Sensitive Editor for OpenVMS Version 4.7.

To associate a user filter with a test, a logical variable starting with the characters "DTM$UF_" is created. The value of the variable is the file specification of the file containing the DECTPU commands. For example:

```
CREATE VARIABLE /LOGICAL DTM$UF_DEVICE "DISK1:[TEST.FILTERS]DEVICE.TPU"
```

The variable is then associated with the test in the usual way, for example:

```
MODIFY TEST /VARIABLE=DTM$UF_DEVICE
```

When the test is run, as part of a collection, the filter will be applied.

2.4.1 New FILTER Command Qualifier

The FILTER command has been extended to include the qualifier /USER_FILTER described below:

```
FILTER

    /USER_FILTER

        /USER_FILTER=filename
        /USER_FILTER=(filename[,...])
        /NOUSER_FILTER (D)
```

The specified files are executed by DEC Text Processing Utility (DECTPU). If more than one file is specified they are executed in the order given. The user filters are applied before any built-in filters that are also specified on the command line.

User filter files can be located either in OpenVMS directories or in Compaq Code Management System for OpenVMS libraries. Files may be specified using logical names that include search lists. Wildcards cannot be used. For files in CMS libraries, the most recent generation on the main line of descent is used.

Before the first file is executed the file to be filtered is read into the DECTPU buffer "filter_buffer". Next, the file specified by the logical name DTM$UFDEFINES is executed. The system logical name DTM$UFDEFINES references the file SYS$LIBRARY:DTM$UFDEFINES.TPU, which contains definitions of a global replace procedure and patterns which can be used in building filters. This logical can be redefined to point to a custom file.

Any errors in accessing the user filter files or in executing the DECTPU commands will be reported. However, they will not cause the FILTER command to fail, and any remaining user and built-in filters will be applied.

After all the user filters have been applied, the file being filtered will be written out. If any built-in filters are also specified, they are applied to the newly created file, resulting in a second new version.

2.4.2 New Variables for User Filters

Variables with names beginning with the string "DTM$UF_" may now be defined. These variables must be logical variables but can be global or local.

When tests are run that have associated variables whose names begin with the string "DTM$UF_", DTM will apply the user filters contained in the files referenced by the value of those variables. Only a single file may be referenced by each variable.

The specified files are executed by DEC Text Processing Utility (DECTPU). If more than one user filter variable is associated with a test, the files are executed in the lexicographic order of the variable names. The user filters are applied before any built-in filters that are also specified for the test.

User filter files can be located either in OpenVMS directories or in CMS libraries. Files may be specified using logical names including logical names that specify search lists. Wildcards cannot be used. For files in CMS libraries, the most recent generation on the main line of descent is used.

Before the first file is executed the file to be filtered is read into the DECTPU buffer "filter_buffer". Next, the file specified by the logical name DTM$UFDEFINES is executed. The system logical name DTM$UFDEFINES references the file SYS$LIBRARY:DTM$UFDEFINES.TPU, which contains definitions of a global replace procedure and patterns which can be used in building filters. This logical can be redefined to point to a custom file.

Any errors in accessing the user filter files or in executing the DECTPU commands will be reported. However, they will not cause the filter operation to fail, and any remaining user and built-in filters will be applied.

After all the user filters have been applied, the file being filtered will be written out. If any built-in filters are also specified, they are applied to the newly created file, resulting in a second new version.

In order to disable a user filter that is defined with a global variable for a particular test, define the value of the variable for the test as a string containing only spaces. For example:

```
MODIFY TEST CHECK_DEVICES /VARIABLE=DTM$UF_DEVICE=" "
```

2.4.3 Filtering Benchmarks on Recording Interactive Terminal Tests

When using the RECORD/FILTER command to filter the benchmark produced by recording an interactive terminal test, user filters associated with the test will be applied, provided that the /VARIABLES qualifier is also used. For example:

```
RECORD /FILTER /VARIABLES TERMINAL_TEST1
```

2.4.4 Using the Global Replace Procedure

The supplied file SYS$LIBRARY:DTM$UFDEFINES.TPU contains a global replace procedure and some predefined patterns that can be used to build filters. The specification of procedure global_replace is as follows:

```
PROCEDURE global_replace ( pattern_to_replace,
                           replacement_string;
                           search_mode,
                           evaluate_replacement,
                           convert_linefeeds)

DESCRIPTION:

  Replace all occurrences of a given pattern with a
  given string in the buffer "filter_buffer".

PARAMETERS:

  pattern_to_replace        The pattern to be replaced.

  replacement_string        The string to be substituted.

  search_mode (optional)    The mode of pattern matching
                            to be used when searching
                            for the pattern. Should be
                            one of:
                              NO_EXACT (default)
                              EXACT
                              TPU$K_SEARCH_CASE
                              TPU$K_SEARCH_DIACRITICAL

  evaluate_replacement      Specifies whether the replacement
    (optional)              string is to be evaluated.
                            Should be one of:
                              OFF, 0 (default)
                              ON, 1
                            If specified as ON or 1,
                            the replacement string is
                            evaluated before use. This is
                            needed if the replacement
                            string contains any partial
                            pattern variables. In this
                            case, any string literals
                            in the replacement string
                            must be specified as nested
                            strings and partial pattern
                            variables converted to
                            strings using STR.
```

```
convert_linefeeds          Specifies whether any linefeed
  (optional)               characters in the replacement
                           string are to be converted
                           into line breaks.
                           Should be one of:
                             OFF, 0 (default)
                             ON, 1
```

2.4.5 Examples of User Filters

Example 1: Simple Use of procedure global_replace

The following example assumes that the disks are named UDISK{n} where {n} is a number, for example UDISK1, UDISK13. This filter replaces such disk names with the string "DISK_NAME":

```
global_replace ( 'UDISK' + number, 'DISK_NAME')
```

The pattern to replace is built from a string literal ('UDISK'), the concatenation operator (+) and the pattern "number" included in the supplied definitions file. The pattern "number" matches a sequence of digits.

The replacement string is the string literal 'DISK_NAME'.

Example 2: Using the Null Pattern

This example uses the supplied "null" pattern with the DECTPU alternation operator to include an optional element in a pattern.

Supposing that, in the previous example, some of the disk names do not include the leading "U", for example DISK7. The following filter replaces disk names with or without the leading "U":

```
global_replace ( ("U"|null) + "DISK" + number, "DISK_NAME")
```

Example 3: Using Pattern Variables

The following example filters dates in the form DD-MMM-YYYY, for example 11-OCT-1999. Because it only filters this one form of date, it is quicker than the built-in date filter which filters many different date formats. It is also not the exact equivalent of the built-in date filter in other respects, for example it treats 37-NOV-0999 as a date, but should be sufficient for most purposes.

```
day := any(" 123") + digit;
month := "JAN" | "FEB" | "MAR" | "APR" |
         "MAY" | "JUN" | "JUL" | "AUG" |
         "SEP" | "OCT" | "NOV" | "DEC";
year := any(digits,4);
date := day + "-" + month + "-" + year;
global_replace( date, "dd-mmm-yyyy");
```

This filter defines the pattern variables "day", "month" and "year" which are then used to define the pattern variable "date" used in the call to global_replace.

The "day" pattern uses the DECTPU function "any" to match either a space or one of the characters "1", "2" or "3", followed by a digit.

The "month" pattern uses the DECTPU pattern alternation operator "|" to specify a list of alternative string literals.

The "year" pattern uses the DECTPU function "any" with the supplied pattern "digits". The "4" parameter indicates that exactly 4 digits are to be matched.

The "date" pattern concatenates these patterns and linking punctuation.

Example 4: Removing Blank Lines

This filter removes blank lines using the DECTPU keywords LINE_BEGIN and LINE_END.

```
global_replace( LINE_BEGIN + LINE_END, '');
```

The LINE_END keyword absorbs the new line.

The above filter only replaces lines containing no characters. The following filter also replaces lines containing only spaces and tab characters:

```
global_replace( LINE_BEGIN + (white_space|null) + LINE_END, '');
```

Example 5: Using Partial Pattern Variables to Retain Context

This example demonstrates how to use surrounding text to identify a string to be replaced without also replacing the surrounding text.

The following filter replaces the month part of a date with the string "mmm". For example, the string "14-OCT-1999" will be replaced by the string "14-mmm-1999":

```
day := any(" 123") + digit;
month := "JAN" | "FEB" | "MAR" | "APR" |
         "MAY" | "JUN" | "JUL" | "AUG" |
         "SEP" | "OCT" | "NOV" | "DEC";
year := any(digits,4);
date := (day + "-"@day_part) + month + ("-" + year@year_part);
global_replace( date, 'str(day_part) + "mmm" + str(year_part)',,ON);
```

The day part of the date and the "-" character are assigned to the partial pattern variable day_part and the year part of the date and preceding "-" assigned to year_part. These partial pattern variables are then included in the replacement string.

When partial pattern variable are used in the replacement string they must be evaluated for each replacement. To do this, set the parameter evaluate_replacement to ON, as shown above.

When the replacement string is to be evaluated, string literals must be nested inside further quotes. This is most easily done by using single quotes for the outer string and double quotes for any nested string literals, or vice-versa. Also, any partial pattern variables must be converted to strings using the DECTPU procedure STR.

Note that including LINE_END in the definition of a partial pattern variable does not have the effect of retaining the line break. See the following example for a resolution of this problem.

Example 6: Using LINE_END for Context

If the search pattern contains LINE_END, the matched line break will be removed, causing the next line to be appended to the current line. To use LINE_END to only provide context for the search, the line break must be reinserted. This is done using the parameter convert_linefeeds.

If the convert_linefeeds parameter is specified as ON, any linefeed characters appearing in the replacement string are removed and the built-in DECTPU procedure SPLIT_LINE is called at the point of the linefeed character.

The following filter replaces any numbers that are the last characters on a line with the string "x":

```
global_replace (number+LINE_END, "x"+lf,,,ON)
```

The "lf" pattern is defined as a linefeed character in the supplied definitions file.

If a LINE_END is included in a partial pattern variable, the line break can be retained by specifying the second optional parameter to the DECTPU STR procedure as a linefeed character, for example:

```
global_replace (number+(LINE_END@sep), '"x"+STR(sep,lf)',,ON,ON)
```

The second parameter to STR specifies the string that line breaks occurring in the first parameter should be converted to. Line breaks are retained by specifying the linefeed character and setting the parameter convert_linefeeds to ON.

Example 7: Using UNANCHOR to Replace Sections

The DECTPU keyword UNANCHOR can be used to replace sections of text delimited by specified strings. The following replaces all text between the strings "/*" and "*/" with the string "/* Deleted */". The text may run across line boundaries:

```
global_replace ( "/*" + UNANCHOR + "*/", "/* Deleted */")
```

Note that while a similar effect is possible using the COMPARE/SENTINEL command, the filter can be applied to individual tests, whereas the /SENTINEL qualifier applies only to collections.

Example 8: Using Other DECTPU Commands

The global_replace procedure can be used for many filtering tasks. However any DECTPU commands can be used to build filters. The file being filtered is read into the buffer "filter_buffer" before the user filters are applied and written out afterwards.

The following filter uses the DECTPU EDIT procedure to convert all characters to uppercase:

```
EDIT( filter_buffer, UPPER, OFF)
```

Note that while a similar effect is possible using the COMPARE/IGNORE=CASE command, the filter can be applied to individual tests, whereas the IGNORE qualifier applies only to collections.

The following filter searches for numbers and replaces them only if they are in a specified range:

```
POSITION (BEGINNING_OF (filter_buffer));
LOOP
    found_range := SEARCH_QUIETLY (number, FORWARD);
    EXITIF found_range = 0;
    POSITION (END_OF(found_range));
    MOVE_HORIZONTAL(1);
    value := INT(STR(found_range));
    IF (value>350) AND (value<570)
    THEN
        COPY_TEXT ("XXX");
        ERASE (found_range);
    ENDIF;
ENDLOOP;
```

The initial POSITION is required to ensure that the whole of the filter_buffer is processed, because the editing point is undefined at the start of each filter. Then, as each number is processed, the editing point is moved to the end of the number. The MOVE_HORIZONTAL procedure call is necessary because the previous POSITION leaves the editing point at the last character of

the number, which would result in an immediate match on the next call to SEARCH_QUIETLY.

3
Problems Fixed

The following problems were fixed and restrictions were removed in DTM Version 4.0.

- ID:49 - Variable View remarks were missing.

  Remarks associated with variables are now displayed in the Variable View.

- ID:364 - Subgroups remarks were missing in Group View.

  Remarks associated with subgroups are now displayed in the Group View.

- ID:404 - SUBMIT/CHARACTERISTICS command failed to set multiple characteristics.

  The SUBMIT/CHARACTERISTICS command now sets all the specified characteristics.

- ID:558 - Character comparison worked incorrectly.

  The character comparison facility no longer stops comparison upon detection of the first difference between a result and benchmark file.

- ID:621 - Use of DTM$OMIT_PRINTABLE_SCREENS caused error messages.

  Using the variable DTM$OMIT_PRINTABLE_SCREENS no longer results in error messages when updating benchmarks for interactive tests.

- ID:623 - Global variables were not defined when recording a test.

  Global variables are now defined when recording a test using the RECORD /VARIABLES command.

- ID:678 - Create and Modify Test dialog boxes did not allow a variable value containing commas.

  The Create Test and Modify Test dialog boxes now allow variables where the variable value contains commas.

- ID:686 - Stop using Tasks dialog box corrupted collection.

Stopping a collection using the Tasks dialog box no longer corrupts the collection.

- ID:693 - Embedded quotes were not valid with /COMMAND qualifier.

  Embedded quotes are now valid with the /COMMAND qualifier for the CREATE TEST_DESCRIPTION and MODIFY TEST_DESCRIPTION commands.

- ID:753 - Help on keypad definitions generated errors.

  Help for keypad definitions is now provided.

- ID:914 - The access violation running interactive tests which use SET HOST in DECwindows has been fixed.

- ID:945 - Creating a collection containing terminal test failed.

  Creating a terminal test, recording it, and then creating a collection containing the test no longer reports that the session file does not exist.

- ID:946 - Collection View remarks were missing.

  Remarks associated with collections are now displayed in the Collection View.

- ID:948 - Updating benchmark held in CMS library sometimes failed.

  Using the DECwindows interface to update a benchmark which is held in a CMS library now works reliably.

- ID:950 - Running collections created /NOVERIFY produced spurious warning message.

  Running a collection that was created with the /NOVERIFY qualifier no longer results in a spurious warning message.

- ID:951 - File name was missing from some CMS error messages.

  When an operation refers to a CMS class and the relevant file is not in the class, the error message produced now contains the name of the file.

- ID:953 - PRINT/BENCHMARK printed the wrong file for an interactive test.

  If the benchmark for an interactive test is held in a CMS library, the PRINT/BENCHMARK command now correctly prints the benchmark screens file (.BMK_SCREENS) rather than the benchmark file (.BMK).

- ID:954 - PRINT/BENCHMARK failed for benchmarks held in CMS library.

  The PRINT/BENCHMARK command, where the benchmark file is held in a CMS library, now works correctly.

- ID:955 - Spurious CMS error messages were produced reviewing collections.

  When reviewing a collection containing new tests in a DTM library where the default benchmark directory is a CMS library, some commands resulted in spurious error messages. These have now been eliminated.

- ID:959 - Recording interactive test from input file failed for comments.

  Recording an interactive test from an input file containing comments now works correctly.

- ID:961 - Errors in collections created /NOVERIFY produced wrong message.

  When errors occur accessing the prologue or template file in a collection created with the /NOVERIFY qualifier, the correct error messages are now produced.

- ID:963 - RUN command hung after certain errors.

  The RUN command no longer hangs. This problem occurred in certain error situations, for example if a template file was read protected, or if a template file was moved or deleted after the collection was created.

- ID:967 - EXTRACT command translated into strings not compatible with device.

  When the EXTRACT command is used on a test originally recorded with VT52 or VT100 terminal characteristics, the generated input file now contains only strings that can be used to rerecord on a corresponding terminal.

- ID:975 - The variable DTM$DATE_FILTER_FIRST could not be created.

  The variable DTM$DATE_FILTER_FIRST can now be created.

- ID:976 - Eight-bit control characters were rejected recording from input file.

  Eight-bit control characters are now accepted when recording an interactive test from an input file.

- ID:978 - Filtering on OpenVMS Alpha Version 7.2 was inconsistent with other systems.

  The directory, filename and version filters on systems running OpenVMS Alpha V7.2 now produce the same results as VAX systems and earlier versions of OpenVMS on Alpha systems. In addition, a number of obscure cases were corrected affecting all systems.

- ID:982 - DECwindows tests ran slower than expected.

DECwindows tests are now run at the correct speed and %DTM-W-TIMEOUTFIRED messages are not reported for strings that have, in fact, occurred.

- ID:985 - Button presses were not properly synchronized when running DECwindows tests.

  When running DECwindows tests, button presses can no longer occur before the preceding synchronization text has been matched.

4
Known Problems

4.1 General

The following known problems exist for DTM Version 4.0:

- ID:645 - Rooted logicals cause failures when tests are compared.

  Rooted logical names are not valid in the specification of test benchmark files. If used, the test will execute successfully but the comparison of the test result file and the benchmark file will fail. The test benchmark file can be amended using the commands: CREATE, MODIFY TEST /BENCHMARK and SET BENCHMARK_DIRECTORY. An example of a rooted logical definition is DISK$:[USER.].

- ID:671 - Drawing using REGIS causes DTM to hang.

  Using the interactive terminal record function to record tests of applications which perform drawing using REGIS causes DTM to hang. This occurs due to a buffer limit inside DTM.

- ID:744 - DTM STOP command can delete the wrong batch job.

  If a collection is submitted to batch and the batch job terminates abnormally, the DTM library will indicate the collection is still running. If the DTM STOP command is specified for this collection, the batch job will be deleted using the entry number for the job. However, the job created for the collection no longer exists, which then allows the queue manager to reuse the entry number. The result is that the wrong batch job can be deleted.

  The DCL commands SHOW QUEUE and/or SHOW ENTRY should be used to verify that the collection is still executing, before using the DTM STOP command. If the collection is not currently executing, the VERIFY /RECOVER command should be used to correct the inconsistency in the DTM library.

- ID:835 - Using /REALTIME qualifier with RECORD/INPUT can result in slow test execution.

Specifying the /REALTIME qualifier on the RECORD /INPUT command to record an interactive terminal test, where the input file contains WAIT records, results in duplication of wait records in the new session file. This causes substantially slower test execution. The workaround is to remove unwanted WAIT records from the session file. See the *Guide to DIGITAL Test Manager for OpenVMS Systems* for information about these topics.

- ID:893 - DECwindows testing problems repositioning windows.

  A test recording containing an action to reposition a window will cause the test to hang and the mouse pointer to change to the plus sign (+) with arrowheads. The test will only continue, from the original reposition action, if the window is repositioned manually and MB1 clicked.

  A similar problem occurs if windows are resized, except the mouse pointer changes to a resize cursor.

  The workaround to the repositioning problem is to set the Workspace Options to "When moving a window, show the entire window" rather than showing the outline only. This works whichever option was applicable when the test was recorded. There is no equivalent workaround for the resizing problem.

- ID:947 - Failure with DECwindows test on OpenVMS Alpha Version 7.x

  The screensave operation for DECwindows tests may fail on systems with OpenVMS Alpha Version 7.1. The problem occurs on systems with some TGA graphics cards.

  When the failure occurs, the following messages are displayed:

  ```
  X Error of failed request: BadColor (invalid Colormap...
    Major opcode of failed request:  91 (X_QueryColors)
    Resource id in failed request:  number
  Serial number of failed request:  number
    Current serial number in output stream:  number
  %XLIB-E-ERROREVENT, error event received from server
  ```

  The failure can occur when the tests are recorded or played. The test will continue, but the screen is not saved.

  The workaround for this problem is to install Compaq Open3D for OpenVMS Alpha Version 4.6. Note that no license is required to install Open3D and use only its 2D graphics capabilities which include a resolution of this problem.

  This problem also occurs on systems with OpenVMS Alpha Version 7.2. The solution is to install the following ECO:

  ```
  OpenVMS VMS72_GRAPHICS-V0100 Alpha V7.2 4D20T Support ECO
  ```

- ID:980 - Missing messages playing DECwindows tests on Alpha systems.

When playing DECwindows tests, either with the PLAY command or as part of a collection, some messages relating to synchronization are not displayed. If the text in a WaitForText record is not matched within the timeout period, the messages %DTM-I-FORCESYNCH and %DTM-W-TIMEOUTFIRED should be displayed, but are not.

In addition, if the "generalVerboseFlag" resource is set to "1", the message %DTM-I-SYNCHMATCH is not displayed when text is successfully matched.

- ID:981 - Details wrong in messages for DECwindows tests on Alpha systems.

  When recording or playing a DECwindows test on Alpha systems, some details in the informational messages are incorrect. The %DTM-I-XTRAPVERSION message displays the wrong version number for XTrap. Also, the %DTM-I-RECORDING and %DTM-I-PLAYING messages display "at line 0" rather than the correct line number.

4.2 DECwindows Interface

The following known problems exist for the DECwindows interface in DTM Version 4.0:

- ID:398 - "^P W" and "^P !" commands for interactive terminal tests do not work.

  When recording interactive terminal tests using the DECwindows interface, the "^P W" and "^P !" commands to enter wait records and comments do not work. If these capabilities are required, record interactive terminal tests from the DCL interface.

- ID:931 - Failure with DECwindows Motif for OpenVMS Versions 1.2-4 and 1.2-5.

  Operations which create a DECterm to display messages may hang when DTM is used with DECwindows Motif for OpenVMS Version 1.2-4 or on VAX systems with DECwindows Motif for OpenVMS Version 1.2-5. After two minutes the hang times out and an error message may be displayed.

  The operations affected include recording and playing interactive terminal and DECwindows tests, running and comparing collections and displaying interactive test result files, benchmark files or difference files.

  The problem arises from a DECwindows Motif defect which prevents the creation of the DECterm. Specifically, DECwindows creates a terminal controller with the wrong language information and therefore the terminal controller never connects with the DTM request to create a terminal.

The problem does not occur if the display used for the DTM windows is running the New Desktop.

The workaround for this problem is to set the "*xnlLanguage" X-resource to the value "NO_LANG". The easiest way to set this resource is to add to the SYS$LOGIN:DECW$XDEFAULTS.DAT file (creating it if need be) the line:

```
*xnlLanguage:   NO_LANG
```

When next logging in to a new session this will take effect.

In a purely English language environment, this workaround should not have any drawbacks. In a multilingual environment, there may be side effects on some applications as their language defaults are overridden. These will need to be determined empirically.

- ID:965 DECterm positioning information ignored by DECwindows Motif for OpenVMS Version 1.2-5.

  The DECterm created by RECORD, PLAY and RUN operations is incorrectly positioned when using DECwindows Motif for OpenVMS Version 1.2-5. Also, the logical names DTM_DECTERM_X and DTM_DECTERM_Y have no effect on the position.

  This is caused by a problem in DECwindows Motif for OpenVMS Version 1.2-5, which ignores the positioning information.

- ID:972 - Failure expanding a collection that is running.

  Expanding a collection, either using the expand attributes item on the view menu or by double clicking on the collection, causes a failure if the collection is running.

  This problem only occurs if the windows are being displayed on a PC using Digital eXcursion.

  If DTM has been started with a DCL command, the following messages are displayed on the terminal where the command was issued:

  ```
  %DTM-F-BUG, There is a bug in Digital Test Manager or something it calls
  -DTM-F-LIBLOCKED, cannot lock the library; already locked
  ```

  The DTM windows disappear and the above messages are produced.

  The workaround is to use the DCL command DTM SHOW COLLECTION.

- ID:973 - The "Hold until" field on the Submit window does not work.

  The "Hold until" field on the Submit window does not work correctly. If a time is specified, the wrong date is used. If only a date is specified, the time part defaults to the current time rather than midnight.

The workaround is to submit collections using the DCL command SUBMIT /AFTER.

4.3 Documentation

The following known problems exist in the DTM Version 4.0 documentation:

- ID:974 - RECORD/VARIABLES incorrectly described.

  In the description of the RECORD command in *DIGITAL Test Manager for OpenVMS Reference Manual*, the default value for the /VARIABLES qualifier should be specified as /NOVARIABLES. Also, the restrictions relating to the /VARIABLE qualifier, which should be the /VARIABLES qualifier, are incorrect. There are no restrictions.

  The /VARIABLES qualifier is omitted from the list of qualifiers in the glossary. The qualifier specifies whether DTM is to define symbols and logical names as specified by the variables associated with the test before recording starts.

- ID:986 - Errors in review subsystem default keypad diagram.

  In *Guide to DIGITAL Test Manager for OpenVMS Systems*, Figure 5-1 Review Subsystem Default Keypad is wrong.

  KP5 issues the INSERT command to mark the test for insertion in a group. This is missing from the diagram.

  PF2 is marked HELP, but really issues the HELP KEYPAD command. GOLD-PF2 issues the HELP command.

  These corrections have been made in the HELP information. To see the corrected diagram, enter the review subsystem and then type HELP KEYPAD or press PF2.

  Table 5-4 which follows the diagram also contains errors:

  The KP7/8/9 keys should be described as "Displays the difference/result /benchmark file".

  KP4 should be described as "Places the files currently selected for printing on the print queue immediately."

  Period and PF1-period display the first/last test, not the first/last screen.

  For PF1-comma, the information in brackets is wrong and should be removed.

  For PF2, the description is wrong. See above for the correct meaning.

  KP5 is the INSERT command. See above for the description.

- ID:987 - Description of DTM$OMIT_PRINTABLE_SCREENS is wrong.

In *Guide to DIGITAL Test Manager for OpenVMS Systems*, the description of the variable DTM$OMIT_PRINTABLE_SCREENS contains several errors.

This variable affects the COMPARE command, which is executed after the individual tests have been run. Consequently, it must be defined as a global variable. Also, it cannot be used to control whether individual tests produce screens files; it only applies to collections.

Further to the above, the manual should note that the variable will only work if the collection is compared as part of the run. For collections created with the /NOCOMPARE qualifier, the same effect can be achieved by defining a logical name before the COMPARE command as follows:

```
$ DEFINE DTM$OMIT_PRINTABLE_SCREENS "1"
$ DTM COMPARE collection_name
```

The example for creating the variable should include the /GLOBAL qualifier as follows:

```
DTM> CREATE VARIABLE /LOGICAL /GLOBAL DTM$OMIT_PRINTABLE_SCREENS "1"
```

The second example for creating a collection using the variable is wrong and should be removed.

5
Restrictions

The following restrictions exist for DTM Version 4.0:

- Use of multiple versions of DTM.

  If different versions of DTM are used on the same DTM library, collections should only be run using the version which last created or recreated a collection.

  When switching to a different version of DTM, always create a collection before any collections are run. Do not create any collections with a different version until all collections have been run to completion.

6
Reporting Problems and Suggestions

Depending on the nature of the request or problem, customers can call Compaq provided their software contract and warranty entitles them to telephone support.

Review the SPD and Warranty Addendum for an explanation of warranty.

If a problems is encountered during the warranty period, customers can report the problem using DSNLink. In the United States, customers can call the Compaq Customer Services Center at 1-800-354-9000.

In addition, Compaq now provides access to Web Information and Support (WIS), an Internet-based electronic services application that allows customers to submit electronic service requests, access databases, download patch kits, transfer files from Compaq, and connect to global notes conferences via the World Wide Web.

Dial-In Access (DIA) users can access WIS by:

1. Opening http://www1.support.compaq.com/wis.
2. Choosing a country code.
3. Entering DIA Access ID and password.

WIS and DIA use the same underlying account and directory on the Compaq Customer Services Center host system.

Customers can register for DIA, and receive a WIS account by contacting their Compaq Customer Services Center or local Compaq representative.

This method can also be used to make requests for additional features or modifications to existing features.

When submitting a report, please include the following information:

1. DTM version and type of OpenVMS operating system.
2. Problem summary.
3. Stack dumps or error logs (if applicable).

4. Files in machine-readable media: magnetic tape (preferred), floppy diskettes, or tape cassettes.