Compaq DECset for OpenVMS

Compaq Code Management System for OpenVMS Version 4.1 Release Notes

August 2000

These release notes for the Compaq Code Management System for OpenVMS Version 4.1 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 Code Management System for OpenVMS, Version 4.1

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 Associated Products 1–2
1.4 Installation Requirements 1–3

2 New Features

2.1 Improved Performance 2–1
2.2 Change Remark on a Reservation 2–1
2.2.1 Routine Format and Arguments 2–2
2.3 Delete with Remove 2–3
2.3.1 Routine Format and Arguments 2–4
2.4 Longer Variant Name Support 2–4
2.4.1 Routine Format and Arguments 2–6
2.4.2 Restrictions 2–6
2.5 Class Difference Command 2–7
2.5.1 Routine Format and Arguments 2–8
2.6 Disable Auto-fetch to LSE 2–12
2.7 Copy a Group or Class 2–12
2.7.1 Routine Format and Arguments 2–13
2.8 Selective Deletion of History Records 2–13
2.8.1 Routine Format and Arguments 2–14
2.9 Insert a Generation Based on a Date 2–14
2.9.1 Routine Format and Arguments 2–15
2.10 Enhancements to the VERIFY Command 2–15
2.11 Message Changes and Additions 2–15
2.11.1 Changed messages 2–16
2.11.2 New messages 2–16

3 Problems Fixed

4 Known Problems

4.1 General 4–1
4.2 DECwindows Problems 4–9
4.3 Documentation Problems 4–10

5 Restrictions

6 Reporting Problems and Suggestions

Preface

These release notes characterize the Version 4.1 release of the *Compaq Code Management System for OpenVMS (CMS)* 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 Compaq Code Management System for OpenVMS 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 specifically to the Compaq Code Management System for OpenVMS, Version 4.1 software. Compaq suggests that you upgrade your environment incrementally with each new release, but there is no technical requirement to do so. It is also recommended that you review the release notes for each version of this software since they capture important release-oriented information and advise you of changes between versions.

1.1 Introduction

Compaq Code Management System for OpenVMS Version 4.1 is a minor release.

1.2 Associated Documents

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

The revised installation guide for all DECset components is:

- *Compaq DECset for OpenVMS Installation Guide*

The other documents associated with CMS are:

- *Guide to DIGITAL Code Management System for OpenVMS Systems*
- *DIGITAL Code Management System Reference Manual*
- *DIGITAL Code Management System Callable Routines Reference Manual*

Also applicable are the general DECset documents:

- *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 CMS 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 provided in PostScript (.PS), HTML (.HTML), and ASCII text (.TXT) formats from the following location:

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

1.3 Associated Products

The Compaq DECset Clients for CMS and MMS software provides access from a desktop personal computer (PC) to the CMS libraries and MMS features residing on OpenVMS Alpha and OpenVMS VAX Systems. Note that the Compaq DECset Clients for CMS and MMS software is packaged and distributed separately from the DECset, CMS, and MMS products.

DECset Clients for CMS and MMS can be used from a personal computer running Microsoft Windows 95, Windows 98, Windows NT, or Windows 2000. Connection to a CMS library from the PC is possible once the server software (provided with the client) and the CMS for OpenVMS software are installed at the server site.

The main DECset Client windows offer menu access to most of the CMS and MMS functions. DECset Client commands can also be performed by clicking the command buttons on the toolbar. In addition, DECset Client supports drag-and-drop functionality for manipulating data within the DECset Client window. For configuration management, a set of files is stored in a CMS library on the OpenVMS system. These files can be fetched, reserved, and replaced in either the PC environment using the CMS Client or the local OpenVMS environment using CMS. As a result, a common configuration can be maintained regardless of the operating environment.

Additional information on DECset Clients for CMS and MMS can be found in the Compaq OpenVMS web pages:

http://www.openvms.digital.com/commercial/decset/clients_index.html

1.4 Installation Requirements

For a new installation of CMS the following settings are required:

OpenVMS Alpha Systems

Global Pages	3500
Global Sections	10
Disk Blocks	44000

OpenVMS VAX Systems

Global Pages	1500
Global Sections	15
Disk Blocks	9000

2
New Features

The following enhancements were added to CMS for the Version 4.1 release.

2.1 Improved Performance

CMS Version 4.0 saw a decline in performance due to the addition of new functionality. This version of CMS has had its performance increased to match the level of CMS Version 3.9.

2.2 Change Remark on a Reservation

The function of the MODIFY command has been extended to enable you to modify aspects of a particular reservation of an element. With the new MODIFY RESERVATION command, you can change the original remark entered for a specific reservation.

The syntax for this command is as follows:

```
MODIFY RESERVATION element_expression [modify_command_remark]
        /REMARK=new_remark_for_reservation (required)
        /GENERATION=generation_expression
        /IDENTIFICATION_NUMBER=n
        /CONFIRM
        /OCCLUDE[=option]
```

The /REMARK qualifier is required for the MODIFY RESERVATION command. The new_remark_for_reservation is a quoted string that will replace the remark currently on the reservation identified by the element expression and the /GENERATION and /IDENTIFICATION_NUMBER qualifiers.

The following sections describe the format and arguments for the MODIFY RESERVATION routine as they pertain to the callable interface.

2.2.1 Routine Format and Arguments

```
CMS$MODIFY_RESERVATION (
     library_data_block,
     element_expression,
     [generation_expression],
     [identification_number],
     [modify_command_remark],
     new_remark_for_reservation,
     [msg_routine])
```

library_data_block
Type: cntrlblk
Access: modify
Mechanism: by reference

Specifies an intialized LDB.

element_expression
Type: char_string
Access: read
Mechanism: by descriptor

Specifies one or more elements or groups of elements whose reservation remarks are to be changed. Wildcards and a comma-separated list are allowed.

You must include a period (.) in the element expression to select one or more elements from the complete list of elements in the library. If you do not include a period, CMS interprets the parameter as a group name and selects elements based on the list of groups established in the library.

generation_expression
Type: char_string
Access: read
Mechanism: by descriptor

Specifies the generation of an element reservation that is to be changed.

identification_number
Type: longword_signed
Access: read
Mechanism: by reference

Specifies the reserved generation of the element whose remark is to be changed. CMS assigns a unique reservation identification number to each element when it is reserved. If an element generation has only one reservation, you can change that reservation remark by specifying the generation expression. However, if multiple reservations exist for the element generation, you must specify the identification number of the exact reservation.

Use the CMS$SHOW_RESERVATIONS routine to determine the reservation number of a generation.

modify_command_remark
Type: char_string
Access: read
Mechanism: by descriptor

Specifies the new remark string to be logged in the history file along with the command.

new_remark_for_reservation
Type: char_string
Access: read
Mechanism: by descriptor

Specifies the remark string to replace the remark currently on the reservation.

msg_routine
Type: procedure
Access: read
Mechanism: by reference

Specifies a message-handler routine. For information about writing a message-handler routine, see Section 1.8 of the *DIGITAL Code Management System Callable Routines Reference Manual*.

2.3 Delete with Remove

This new qualifier to the DELETE command for classes and groups combines the DELETE command with the REMOVE command. It first removes any entries in the class or group, then deletes the class or group itself.

The syntax for this command is as follows:

```
DELETE GROUP/REMOVE_CONTENTS group_expression
DELETE CLASS/REMOVE_CONTENTS class_expression
        /LOG
        /CONFIRM
        /OCCLUDE[=option]
```

2.3.1 Routine Format and Arguments

```
CMS$DELETE_CLASS (
     library_data_block,
     class_expression,
     [remark],
     [msg_routine],
     [remove_contents])

CMS$DELETE_GROUP (
     library_data_block,
     group_expression,
     [remark],
     [msg_routine],
     [remove_contents])
```

remove_contents
Type: longword_signed
Access: read
Mechanism: by reference

Determines whether CMS removes the contents of the group or class before the deletion. The default value of False (0) tells CMS not to delete the contents. Specify a value of True (1) to have CMS first remove the contents of the group or class before deleting it.

For complete information on the CMS$DELETE_CLASS and CMS$DELETE_GROUP routines, see the *DIGITAL Code Management System Callable Routines Reference Manual*.

2.4 Longer Variant Name Support

Variant names are no longer limited to a single character. This facility has been expanded to allow a string of 255 alphabetic or underscore (_) characters. To use longer variant names, supply the long name to the REPLACE/VARIANT command.

A new library attribute LONG_VARIANT_NAMES has been added. Before generations with long variant names can be replaced in a library, the LONG_VARIANT_NAMES attribute must be set for the library.

The LONG_VARIANT_NAMES attribute can be set using the /LONG_VARIANT_NAMES qualifier with the CREATE LIBRARY and MODIFY LIBRARY commands, and reset using the /NOLONG_VARIANT_NAMES qualifier. In the CMS DECwindows interface, the New Library and Modify Library dialog boxes also provide for control of the this attribute.

The syntax for these commands is as follows:

```
CREATE LIBRARY directory-specification
        /AFTER[=directory-specification]
        /BEFORE[=directory-specification]
        /[NO]CONCURRENT
        /[NO]CONFIRM
        /[NO]CREATE_DIRECTORY
        /[NO]EXTENDED_FILENAMES
        /[NO]KEEP
        /[NO]LOG
        /[NO]LONG_VARIANT_NAMES
        /REFERENCE_COPY=directory-specification
        /NOREFERENCE_COPY
        /REVISION_TIME[=option]
```

/LONG_VARIANT_NAMES

Used to create a CMS library that allows the use of long variant names. Use /NOLONG_VARIANT_NAMES to create a CMS library that does not allow the use of long variant names. /NOLONG_VARIANT_NAMES is the default.

```
MODIFY LIBRARY
        /[NO]CONCURRENT
        /[NO]CONFIRM
        /[NO]EXTENDED_FILENAMES
        /[NO]KEEP
        /[NO]LOG
        /[NO]LONG_VARIANT_NAMES
        /OCCLUDE[=option,...]
        /REFERENCE_COPY=directory-specification
        /NOREFERENCE_COPY
        /REVISION_TIME[=option]
```

/LONG_VARIANT_NAMES

Used to change the attribute of a CMS library to allow the use of long variant names. Use /NOLONG_VARIANT_NAMES to change the attribute to disallow the use of long variant names. If this qualifier is not specified the status of the attribute remains unchanged.

2.4.1 Routine Format and Arguments

```
CMS$CREATE_LIBRARY (
    library_data_block,
    directory,
    [remark],
    [reference_copy_dir],
    [msg_routine],
    [confirm_routine],
    [output_routine],
    [width],
    [position],
    [positional_dir_spec],
    [revision_time],
    [auto_create],
    [concurrent],
    [0],
    [keep],
    [extended_filenames],
    [long_variant_names])

CMS$MODIFY_LIBRARY (
    library_data_block,
    [remark],
    reference_copy_dir,
    [msg_routine],
    [revision_time],
    [concurrent],
    [0],
    [keep],
    [extended_filenames],
    [long_variant_names])
```

long_variant_names
Type: mask_longword
Access: read
Mechanism: by reference

The default 0 does not allow long variant names. The value 1 allows long variant names.

2.4.2 Restrictions

If you attempt to use an earlier version of CMS (prior to Version 4.1) to access to a library that has the long variant name attribute set, the following error message is displayed:

```
%CMS-F-BADLIB, there is something wrong with your library
* CMS-F-BADTYPSTR, header block type is 145; it should be 17
```

This will occur for any command issued against the library by an older version of CMS until a CMS Version 4.1 user successfully resets the attribute on the library.

2.5 Class Difference Command

CMS now has the ability to compare two classes in several different ways. The differences are defined by using the /SHOW qualifier with a keyword that denotes the type of differences to display.

If the particular generation of any element, that is a member of the class, differs between the classes, CMS creates a file that contains a listing of the class members that differ between them. If the contents of the classes are the same, CMS issues a message to that effect and does not create a differences file (unless the /FULL qualifier is in effect). The /CLASS qualifier must be specified prior to any other qualifiers.

The syntax for this command is as follows:

```
DIFFERENCES/CLASS class-name1 class-name2
               /APPEND
               /FULL
               /IGNORE=FIRST_VARIANT
               /LOG
               /OCCLUDE[=options]
               /OUTPUT[=file_specification]
               /PARALLEL
               /SHOW=(options[,...])
               /WIDTH=n
```

/IGNORE=FIRST_VARIANT

Directs CMS to ignore any differences where the generation in one class is the first variant of the generation in the other class. For example, generation 2C1 is a first variant of generation 2 and generation 4B6D1 is a first variant of generation 4B6.

/SHOW=(keyword[,...])

Specifies one or more of the following keywords. Each keyword indicates a type of class difference to show.

GENERATION_DIFFERENCES - Directs CMS to show the difference if the generation in one class differs from the generation in the other class but one generation isn't a variant of the other generation.

VARIANT_DIFFERENCES - Directs CMS to show the difference if the generation in one class differs from the generation in the other class and one generation is a variant of the other. One generation will be a variant of the other if at some point in the path between one generation and the other a generation is reserved and replaced /VARIANT.

If the /SHOW qualifier is not specified, both generation differences and variant differences are shown. If a generation of an element is in one class but no generation of the element is in the other class then the difference is always shown regardless of whether the /SHOW qualifier is specified or not.

The /CLASS qualifier is required. See the *DIGITAL Code Management System Reference Manual* for information on other qualifiers and parameters.

2.5.1 Routine Format and Arguments

```
CMS$DIFFERENCES_CLASS (
            library_data_block,
            class_expression1,
            class_expression2,
            [append],
            [format],
            [full],
            [ignore_mask],
            [nooutput],
            [parallel],
            [show_mask],
            [width],
            [output_file],
            [output_routine],
            [user_arg],
            [msg_routine])
```

All arguments not described below have the same functionality as in the CMS$DIFFERENCES routine. See the *DIGITAL Code Management System Callable Routines Reference Manual* for details.

class_expression1
Type: char_string
Access: read
Mechanism: by descriptor

Specifies the first class name to be used for the differences.

class_expression2
Type: char_string
Access: read
Mechanism: by descriptor

Specifies the second class name to be used for the differences.

ignore_mask
Type: mask_longword
Access: read
Mechanism: by reference

Specify whether CMS ignores the first variant of a generation. To treat the first variant as a difference, specify a value of 0. Use the constant CMS$M_ IGNORE_FIRST_VARIANT to ignore the first variant.

Directs CMS to ignore any differences where the generation in one class is the first variant of the generation in the other class. For example generation 2C1 is a first variant of generation 2 and generation 4B6D1 is a first variant of generation 4B6.

show_mask
Type: mask_longword
Access: read
Mechanism: by reference

Specifies one or more values for the **show_mask** argument. Values are predefined by using the following mask constants.

CMS$M_ELEMENT_DIF - Include differences where an element is in one class but not the other.

CMS$M_GENERATIONAL_DIF - Include differences where one member generation differs from the other by more than just its variant. Directs CMS to show the difference if the generation in one class differs from the generation in the other class but one generation isn't a variant of the other generation.

CMS$M_VARIANT_DIF - Include differences where one member generation differs from the other by its variant. Directs CMS to show the difference if the generation in one class differs from the generation in the other class and one generation is a variant of the other. A generation will be a variant of another if at some point in the path between one generation and the other a generation is reserved and replaced /VARIANT.

These values may be ORed together to allow combinations of these values to be used. If a NULL parameter or 0 is supplied the default is to show ELEMENT, GENERATIONAL and VARIANT differences.

output_routine
Type: procedure
Access: read
Mechanism: by reference

The unformatted callback routine for CMS$DIFFERENCES_CLASS( ) is called with the following parameters:

```
diff_class_unformatted_callback(first_call,
                                library_data_block,
                                user_param,
                                eof_flag,
                                diff_flag,
                                output_record_id1,
                                output_record_id2,
                                entry_number1,
                                entry_number2)
```

Parameters are all as documented for the unformatted callback routine for CMS$DIFFERENCES( ) in the *DIGITAL Code Management System Callable Routines Reference Manual* except where noted below:

eof_flag
Type: longword_signed
Access: read
Mechanism: by reference

Same as described in *DIGITAL Code Management System Reference Manual* except that the penultimate sentence should read as follows:

When **eof_flag** is true, the contents of **diff_flag**, **output_record_id1**, **output_record_id2**, **entry_number1**, and **entry_number2** are undefined.

diff_flag
Type: longword_signed
Access: read
Mechanism: by reference

Specifies whether the two records are different or not.

If **diff_flag** is false (0), the two records are the same, but textually they may actually be different. This happens if a full listing is requested and there are generation members that differ between the two classes but are not flagged differently due to the options selected on the **show_mask** and **ignore_mask** arguments to CMS$DIFFERENCES_CLASS.

If **diff_flag** is true (1), **output_record_id1/entry_number1** and **output_record_id2/entry_number1** describe generation members for the same element but whose generation number differs between the two classes. In addition if one of the entry numbers (**entry_number1** or (**entry_number2**) is -1 it means that this element exists in one class but not the other.

output_record_id1
Type: address
Access: read
Mechanism: by reference

Specifies a string identifier for the generation member being passed from CMS$DIFFERENCES_CLASS( ) for class1. This parameter is only valid if **entry_number1** contains a value other than -1. The string has the format:

element-name(generation-name) "generation comment"

If both **output_record_id1** and **output_record_id2** are supplied on the same call they will always identify the same element, albeit different generations of that element.

Use the CMS$GET_STRING routine to translate the string identifier.

output_record_id2
Type: address
Access: read
Mechanism: by reference

Specifies a string identifier for the generation member being passed from CMS$DIFFERENCES_CLASS( ) for class2. This parameter is only valid if **entry_number2** contains a value other than -1. The string has the format:

element-name(generation-name) "generation comment"

If both **output_record_id1** and **output_record_id2** are supplied on the same call they will always identify the same element, albeit different generations of that element.

Use the CMS$GET_STRING routine to translate the string identifier.

entry_number1
Type: longword_signed
Access: read
Mechanism: by reference

Specifies the position of the generation member in the list of generation members that are members of class1. The generation members are labelled from 1 - n.

Note that if particular elements are members of one class but not the other then the element will have a different entry number in each class. This parameter is -1 if no entry from class1 is being specified by this call, in other words this element is in class2 but not in class1.

entry_number2
Type: longword_signed
Access: read
Mechanism: by reference

Specifies the position of the generation member in the list of generation members that are members of class2. The generation members are labelled from 1 - n.

Note that if particular elements are members of one class but not the other then the element will have a different entry number in each class. This parameter is -1 if no entry from class2 is being specified by this call, in other words this element is in class1 but not in class2.

2.6 Disable Auto-fetch to LSE

CMS now provides a setting in the DECwindows interface that lets you specify whether a FETCH or RESERVE operation automatically opens an Compaq Language-Sensitive Editor for OpenVMS buffer for the element.

From the OPTIONS pull-down menu, choose LSE Views. Select Fetch, Reserve, both or neither to have CMS open LSE buffers or not for Fetched or Reserved elements.

2.7 Copy a Group or Class

The Copy command has been expanded to include the capability of copying groups and classes within and between CMS libraries.

The syntax for this command is as follows:

```
COPY class_name new_class_name "remark"
COPY group_name new_group_name "remark"
       /LIBRARY=library_specification
       /LOG
       /CONFIRM
       /OCCLUDE[=option]
```

When copying between libraries, all the elements and generations must exist in the library to which the group or class is being copied.

2.7.1 Routine Format and Arguments

```
CMS$COPY_CLASS (
     library_data_block,
     input_class_expression,
     ouput_class_expression,
     [remark],
     [source_library_data_block],
     [msg_routine])

CMS$COPY_GROUP (
     library_data_block,
     input_group_expression,
     ouput_group_expression,
     [remark],
     [source_library_data_block],
     [msg_routine])
```

The parameters and return codes for CMS$COPY_CLASS and CMS$COPY_ GROUP are the same as the CMS$COPY_ELEMENT routine. See the *DIGITAL Code Management System Callable Routines Reference Manual* for details.

2.8 Selective Deletion of History Records

New qualifiers have been added for the DELETE HISTORY command to allow the deletion of history records

The syntax for this command is as follows:

```
DELETE HISTORY "remark"
        /OBJECT=object_name
        /TRANSACTIONS=(transaction[,...])
        /USER=username
```

All of the qualifiers shown above are new qualifiers for the DELETE HISTORY command and allow the deletion of history records based on the type of transaction, user name, or object name.

2.8.1 Routine Format and Arguments

```
CMS$DELETE_HISTORY (
     library_data_block,
     [remark],
     before,
     [transaction_mask],
     [output_routine],
     [user_arg],
     [msg_routine],
     [object],
     [user])
```

object
Type: char_string
Access: read
Mechanism: by descriptor

A string containing the name of the object whose history is to be deleted.

user
Type: char_string
Access: read
Mechanism: by descriptor

A string containing the name of the user whose history is to be deleted.

For complete information on the CMS$DELETE_HISTORY routine, see the *DIGITAL Code Management System Callable Routines Reference Manual*.

2.9 Insert a Generation Based on a Date

A new qualifier has been added to the INSERT GENERATION command. The qualifier is /BEFORE, which allows the user to specify a date instead of a generation number for inclusion into a class. The qualifier uses the date to determine which generation is inserted. The latest generation before the specified date is used.

The syntax for this command is as follows:

```
INSERT GENERATION element_expression class_name "remark"
        /BEFORE[=date]
```

2.9.1 Routine Format and Arguments

```
CMS$INSERT_GENERATION (
     library_data_block,
     element_expression,
     class_expression,
     [remark],
     [generation_expression],
     [always],
     [supersede],
     [if_absent],
     [msg_routine],
     [before])
```

before
Type: date_time
Access: read
Mechanism: by reference

Specifies a binary date and time value that CMS uses to select the generation for insertion into a class.

For complete information on the CMS$INSERT_GENERATION routine, see the *DIGITAL Code Management System Callable Routines Reference Manual*.

2.10 Enhancements to the VERIFY Command

The VERIFY/REPAIR command has been enhanced such that the following conditions can be repaired when detected in a CMS library.

- Inconsistent class back pointer list on a generation
- Inconsistent free blocks
- Inconsistent class back pointer list entry on a particular generation block
- Inconsistent free block count
- Inconsistent free block list end pointer
- Lost free block

2.11 Message Changes and Additions

The following messages have been added or changed in CMS.

2.11.1 Changed messages

- COPIED - ’operation-type’ ’library’’object-name’ copied to ’library’’object-name’
- COPIES - ’number’ copy operation(s) completed
- ERRCOPIES - ’number’ copy operation(s) completed and ’number’ error(s) occurred

2.11.2 New messages

- BADLST - ’object’ list for generation ’element’(’number’) is corrupted

  This error message is the result of CMS discovering an inconsistency or error in an internal list for the specified generation.

  Use VERIFY/REPAIR, if it is unable to correct the problem, restore a backup copy of your library.

- BADREFHDR - reference copy file ’name’ not closed by CMS

  This error message is the result of CMS discovering that a reference copy file was not closed by CMS. Someone has used something other than CMS to modify the file header information of this reference copy.

  Use VERIFY/REPAIR to recreate the reference copy for the element.

- DIFFCLASS - classes are different

  This message is the result of the DIFFERENCES/CLASS command. It is displayed when differences are found between the two classes. This is an informational message.

- ERRVERGEN -generation ’name’ verified with errors

  CMS encountered one or more errors while verifying a generation in your library.

  Use VERIFY/REPAIR, if it is unable to correct the problem, restore a backup copy of your library.

- FREBLKCON - ’count’ free blocks consolidated into other free blocks

  During a VERIFY/REPAIR operation CMS has consolidated free blocks in the library into other free blocks in order to improve future efficiency. This is an informational message.

- IDENTCLASS - classes are identical

The two classes compared by CMS are identical. This is an informational message.

- INCLIBVER - library version is 'num', maximum supported version is 'num'

  This error message is displayed when an attempt is made to access a CMS library with a database version number that was greater than the maximum version supported by this version of CMS. Upgrade CMS to a version that understands the format of the library.

- LONGVARFOUND - generation 'name' contains a long variant name

  This error message is the result of a failed MODIFY LIBRARY command. CMS has encountered a generation with a long variant name in the library during an attempt to change the library LONG_VARIANT_NAMES attribute from allow or disallow.

  Use DELETE GENERATION to delete this generation and any other generation that contains a long variant name.

- NOBCKPTR - 'object' 'name' has no back pointer to 'object' 'name'

  This error message is the result of CMS encountering an internal error in the library control structure.

  Use VERIFY/REPAIR, if it is unable to correct the problem, use a backup copy of your library.

- NODELACCESS - you do not have DELETE access to the element

  This error occurs if an element cannot be deleted due to the protection on the element file or an ACL preventing access to the element. Add DELETE access to the file associated with the element or change the ACL to allow DELETE access.

- NOGENBEFORE - no generation exists from before 'date'

  The generations of the specified element were all created after the date specified on the /BEFORE qualifier. Correct the date to include the date of a generation.

- NOREPBCKPTR - 'object' name, backpointer to 'object' 'name' not repaired

  CMS was unable to repair the specified backpointer list. Issue another VERIFY/REPAIR operation because later actions of the first VERIFY /REPAIR may make the list repairable.

- REPBADLST - 'object' list for generation 'name' repaired

  CMS has successfully repaired the indicated condition in your library. This is an informational message.

- REPBADTYP - 'object' block type repaired

  CMS has successfully repaired the indicated condition in your library. This is an informational message.

- REPBCKPTR - 'object' name, backpointer to 'object' 'name' repaired

  CMS has successfully repaired the indicated condition in your library. This is an informational message.

- REPCNTSTR - 'object' count repaired

  CMS has successfully repaired the indicated condition in your library. This is an informational message.

- REPENDPTR - 'object' list end pointer repaired

  CMS has successfully repaired the indicated condition in your library. This is an informational message.

- REPMISBLK - a type 'object' missed block repaired

  CMS has successfully repaired the indicated condition in your library. This is an informational message.

- TRUNCLST - 'object' list for generation 'element'('number') truncated

  CMS had to truncate an internal list during a VERIFY/REAPIR operation. This is an error message.

  Use VERIFY/REPAIR again, it should be able to correct the problem. If it is unable to correct the problem, restore a backup copy of your library.

3
Problems Fixed

The following problems were fixed and restrictions removed in CMS Version 4.1:

- ID:50 - DECwindows element view with search-listed libraries incorrectly caused all the elements in subsequent libraries to be displayed.
- ID:51 - DECwindows element view had a problem with the vertical scroll bar. When an element view expanded beyond the confines of the window, for example after expanding all the details of on a library (View - Expand - All), the vertical scroll failed to function. The size of the default view window has been expanded to insure that the scroll bar has a scrollable area.
- ID:53 - `REMOVE GENERATION/IF_PRESENT` returned the wrong status. If `REMOVE GENERATION/IF_PRESENT` was used to remove a generation that did not exist from a class, the status %X10000000 was returned instead of CMS$_NORMAL.
- ID:55 - Elements could not be fetched from the DECwindows class view. This problem was causing an error whether or not the element existed.
- ID:58 - Attempts to fetch or reserve groups or classes using CMS DECwindows interface were causing LSE to report a file not found error.
- ID:66 - CMS$DIFFERENCES would fail if the element was not in the first library of a search list.
- ID:99 - Modify Library has a text widget to hold a library name. The library name was not written into this field.
- ID:110 - In CMS DECwindows, when the CANCEL button was selected to cancel the CLOSE LIBRARY action, the action was not stopped and the library was closed.
- ID:115 - The SHOW HISTORY command was case sensitive. This required the user to type the object in the same case as it was in the history file.

- ID:118 - The CMS message READONLY text was not correct. The CMS READONLY error occurs when elements are not allowed DELETE access. The message for the error now reflects that error correctly.
- ID:120 - When copying elements between two libraries with extended filename support, some elements with escape characters in their names failed to copy.
- ID:121 - After an automatic recovery, messages were being incorrectly generated.
- ID:122 - Using CMS$CREATE_LIBRARY with the **extended_filenames** parameter set to true on a non-ODS-5 disk did not return the appropiate error.
- ID:123 - On OpenVMS VAX systems, VERIFY and VERIFY/REPAIR failed to correctly identify corrupted reference copy files. If the reference copy contained extra lines at the end, CMS would fail to see that the file was corrupt.
- ID:125 - CMS could fail to replace an element that is not in the first library of a search list.
- ID:127 - CMS DECwindows did not honor the Keep/NoKeep library attribute. When the library wide default was NoKeep, a REPLACE operation with the Keep option set, would deleted the file.
- ID:128 - Ctrl/C did not interrupt the DIFFERENCES command.
- ID:133 - When using the DIFFERENCES command, if the second file name is too long, the error message always shows the first file name as being too long.
- ID:137 - When CMS$REMOVE_GROUP was called twice, where the first call was successful and the second was not, CMS could return an incorrect error code.
- ID:140 - Fetching of binary files to a directory without write access caused an unrecoverable error.
- ID:142 - Incomplete NOACCESS error message for DELETE GENERATION and MODIFY GENERATION commands.
- ID:147 - Messages from CREATE LIBRARY were misleading as to why a library could not be created.
- ID:148 - When CMS$INSERT_GROUP was called twice, where the first call was successful and the second was not, CMS could return an incorrect error code.

- ID:150 - VERIFY did not check or repair modification times of reference copies.

  This also corrects the problem with an MMS build that relies on the time of the reference copy.

- ID:157 - The OCCLUDE... sub-dialog from dialogs had no effect and changing the default occlusion only took effect from the second dialog invocation.

  OCCLUDE can now be managed from either the sub-dialog or default occlusion with the desired effect.

- ID:158 - The CREATE ELEMENT command should not have accepted the /OCCLUDE qualifier.

- ID:163 - The CREATE LIBRARY callable interface routine should also set library.

  CMS$CREATE_LIBRARY now calls CMS$SET_LIBRARY.

- ID:167 - In CMS$DIFFERENCES and CMS$DIFFERENCES_CLASS, an output file should not have been required when the **full** argument and an output routine are specified. Also in CMS$DIFFERENCES and CMS$DIFFERENCES_CLASS, if unformatted output is specified and an output_routine is given then the meaningless qualifier **parallel** should have been rejected.

4
Known Problems

The following known problems exist for CMS Version 4.1:

4.1 General

- ID:11 - CMS-E-BADREF, reference copy for element.

  This problem occurs when an element is created from a file that is either indexed or fixed length with the History attribute set AND a reference copy directory defined. CMS attempts to log the history for the element, but since it is fixed, CMS cannot write to the file. At the time of creation, CMS issues a message that the History will not be included in the element, but it fails to reset the attribute. This presents the reference copy as bad when it is not.

  To workaround this problem, either set the element attribute to `NOHISHTORY` or `/NOREFERENCE`.

- ID:14 - CMS differences on VFC files displays incorrect line numbers.

  An attempt to perform a comparison of two VFC files using the `DIFFERENCES` command can generate incorrect line numbering in the differences output. As the files are different, a difference file is created, but the line numbers in the difference file are not correct. Instead of the correct line number, the line number is 36097/1. The problem occurs only with VFC files.

- ID:15 - CMS incorrectly reports a conflict during variant merge operation.

  In certain circumstances, where there are a large number of generations and deleted lines in a relatively small region of a large CMS element file, the comparison algorithm fails to correctly track the deletion information in the element during merges. This can cause attempts to merge a variant (gen 28p1) and the main line of descent (gen 31) with a common ancestor (gen 28) via `FETCH/MERGE` or `RESERVE/MERGE` resulting in a bogus first conflict. The lines in the conflict do not match the lines that are different.

- ID:23 - `DIFFERENCES/SENTINEL` on identical files can find differences.

Using the DIFFERENCES/SENTINEL command to compare identical files incorrectly reports differences. Using the /SENTINEL qualifier with the CMS DIFFERENCES command can cause CMS to report differences in two identical files.

- ID:25 - CMS performs merges operations incorrectly.

  Deleted lines may not stay deleted in a merge operation.

- ID:26 - The CMS MERGE command finds conflicts inconsistently.

  MERGE may produce different results depending on the order in which new generations or variants are inserted into the CMS library. For example, the same second generation and variant are placed into a CMS library but in different order. In the first case, the files are merged; in the second case they are merged with conflicts.

- ID:34 - Problem with FETCH/GENERATION=no *.* and ACLs.

  Errors may be reported while fetching a class containing some elements that are protected by CMS ACLs. For example, where one of the elements has a CMS ACL that states that the user has no access to the element and another element has no ACL, the class is fetched with no privileges turned on. The one element is fetched as expected, but CMS attempts to fetch the other element and the following errors occur:

  ```
  %CMS-E-NOFETCH, error fetching element DISK:[DIR]BOGUS.DAT
  -CMS-E-NOACCESS, no FETCH access to element BOGUS.DAT
  ```

  When BYPASS privilege is turned on, the errors do not occur.

- ID:40 - RMS-E-ISI errors performing wildcard CREATE ELEMENT with mixed files.

  Wildcarded CREATE ELEMENT commands can fail if non-DDIF files follow DDIF files. The following errors are produced:

  ```
  %RMS-E-ISI, invalid internal stream identifier (ISI) value
  -RMS-E-EXTNOTFOU, RMS Extension not found
  %CMS-F-BUG, there is something wrong with CMS ...
  -CMS-F-SYSFAIL, LIB$FREE_VM system service request failed
  -LIB-F-BADBLOSIZ, bad block size
  ```

  DDIF files later in the list will be created without errors.

  The only known workaround is to avoid a wildcard CREATE of non-DDIF files after DDIF files.

- ID:60 - Review state altered after REPLACE/IF_CHANGED is issued on unchanged elements.

  The review state of element generations is incorrectly altered after a REPLACE/IF_CHANGED of an unmodified element.

- ID:70 - Merging mainline generation into variant produces different result:

  ```
  CMS> FETCH element-name/GENERATION=2x1/MERGE=3 ""
  ```

  In relatively rare cases merging a mainline generation into a variant will produce an invalid and different result from merging the same variant into the same mainline generation:

  ```
  CMS> FETCH element-name/GENERATION=3/MERGE=2x1 ""
  ```

  When particular lines have been removed in the mainline generation, in a variant of an earlier generation, and then in an unrelated variant of the common ancestor (generation 2 for the previous example), but not in the common ancestor, the lines are incorrectly included in the merged result when the mainline generation is merged into the variant.

  ```
  *3D        <--- deleted in mainline generation
  *1V1D      <--- deleted in an earlier variant
  *2V1D      <--- deleted in unrelated variant of ancestor
      <--- the deleted line
  *2V1E      <--- delete is the ONLY change (i.e. no inserts)
  *1V1E
  *3E
  ```

  When the variant is merged into the mainline generation, or the generations are created in different order then the lines remain deleted.

  ```
  *3D        <--- deleted in mainline generation
  *2V1D      <--- deleted in unrelated variant of ancestor
  *1V1D      <--- deleted in an earlier variant
      <--- the deleted line
  *1V1E
  *2V1E      <--- delete is the ONLY change (i.e. no inserts)
  *3E
  ```

  The problem can be avoided by always merging the variant in the mainline generation. For example:

  ```
  CMS> FETCH element-name/GENERATION=3/MERGE=2x1 ""
  ```

  Or:

  ```
  CMS> FETCH element-name/MERGE=2x1 ""
  ```

- ID:71 - CMS cannot parse the `/FORMAT` string when it contains a plus sign (+).

```
$ CMS SHOW ELEMENT *.* /FORMAT="$ cms diff #e/gen=1  #e" -
_$ /OUTPUT=differ.com    ! Works

$ CMS SHOW ELEMENT *.* /FORMAT="$ cms diff #e/gen=1+  #e" -
_$ /OUTPUT=differ.com    ! Fails
%CMS-E-ERRPAREXP, error parsing element expression
-CMS-E-ILLCHAR, illegal character in group name
```

- ID:72 - /FORMAT splits long lines.

  CMS splits long output lines. For example:

  ```
  $ CMS SHOW ELEMENT *.* /OUTPUT=sys$scratch:x.tmp -
  _$ /FORMAT="$ IF F$SEARCH(""#e"") .NES. """" THEN -
  _$ CMS DIFFERENCE #e/GEN='p1 #e"
  ```

  If the library contains elements with long names, the output lines for these files can also be split by CMS.

- ID:73 - CMS ignores commands starting with an at sign (@).

  CMS ignores any commands that begin with the at sign (@). For example:

  ```
  $ CMS
  CMS> @x.x
  CMS>
  ```

  No error messages are returned.

- ID:79 - CMS may not be able to parse a /SINCE date-time string containing a plus sign (+).

  CMS may report an unrecognized keyword error in the /TRANSACTIONS qualifier of the SHOW HISTORY command if a plus sign (+) is used after the date-time string of the /SINCE or /BEFORE qualifier. For example:

  ```
  CMS> SHOW HISTORY -
  _CMS> /TRANSACTIONS=(CREATE)/BEFORE="15-AUG-1996 + 00:00:00"
  %CLI-W-IVKEYW, unrecognized keyword - check validity ...
  \(CREATE)\
  ```

  Although the problem only occurs when the /TRANSACTIONS keywords are specified within parentheses, it is triggered by the plus sign (+) in the date-time string.

  The problem can be avoided by removing the plus sign (+) from the date-time string:

  ```
  CMS> SHOW HISTORY -
  _CMS> /TRANSACTIONS=(CREATE)/BEFORE="15-AUG-1996 00:00:00"
  ```

- ID:81 - Quoted generation strings are processed incorrectly.

  When using /GENERATION=generation-expression, do not enclose the generation expression in quotes.

- ID:85 - History and Notes problems on files with MRS attribute.

  CMS fails to recognize the maximum record size (MRS) attribute for Notes and History. As a result, CMS may add Notes or History to a record beyond its maximum size. On FETCH or RESERVE, CMS fails with the following error.

  ```
  CMS-F-BUG, there is something wrong with CMS ...
  CMS-F-NOQIO, $QIO failed
  RMS-F-RSZ, invalid record size
  ```

- ID:94 - When a group and an element have almost the same name, for example group FOO and element FOO.BAR, the command CMS SHOW HISTORY F*.BAR will not resolve the wildcard correctly.

  The command will show the history for both the group and element.

- ID:95 - If the remark field of the CMS MODIFY GENERATION command contains the exclamation point (!), CMS generates the following message:

  ```
  %CMS-F-BUG, there is something wrong with CMS ...
  -CMS-F-SYSFAIL, $FAOL system service request failed
  -SYSTEM-F-BADPARAM, bad parameter value
  ```

- ID:96 - If CMS is unable to delete the input file after a CREATE ELEMENT or REPLACE operation, no message will be issued to inform the user of the problem.

  The element is created or replaced without the file being deleted.

- ID:97 - When using the SET ACL/DELETE command to delete a list of ACEs from an ACL, the operation will terminate if it encounters an identifier that does not exist on the system.

  The correct behavior is to continue the operation with the remaining ACEs specified.

- ID:98 - Certain qualifiers that should only accept parameters in their non-negated form are also accepting them in their negated form, and the qualifier is then treated as if it were not negated. For example, the qualifier /NOHISTORY="#H" should be disallowed, but is being treated as /HISTORY="#H".

- ID:104 - Using the /BEFORE qualifier on a library that was previously set returns both an error and success status without changing the position of the library in the search list.

```
$ CMS SET LIBRARY [A], [B], [C]
%CMS-I-LIBIS, library is DISK$USER:[A]
%CMS-I-LIBINSLIS, library DISK$USER:[B] inserted at end of
     library list
%CMS-I-LIBINSLIS, library DISK$USER:[C] inserted at end of
     library list
%CMS-S-LIBSET, library is set
$CMS SHOW LIBRARY
Your CMS library list consists of:
   DISK$USER:[A]
   DISK$USER:[B]
   DISK$USER:[C]
$ CMS SET LIBRARY [C]/BEFORE
%CMS-E-LIBALRINLIS, library DISK$USER:[C] is already in the
       library list
%CMS-S-LIBSET, library set
Your CMS library list consists of:
   DISK$USER:[A]
   DISK$USER:[B]
   DISK$USER:[C]
```

- ID:113 - CMS Callable Interface mishandles bad boolean values. When a parameter is passed to the Callable Interface whose value should be a boolean, if the value is not 0 or 1, CMS may fail.
- ID:126 - When a replaced generation has no changes CMS fails to update the date attributes of the reference copy.
- ID:141 - CMS$SET_LIBRARY fails if the directory parameter does not contain valid information.
- ID:143 - CMS$SHOW_ELEMENT with a bad parameter in the element name signals a fatal error before calling the output routine.
- ID:144 - A call to CMS$SET_LIBRARY without a CMS license in place signal a fatal error before calling the output routine.
- ID:145 - Calling CMS$INSERT_ELEMENT with a group_expression or remark parameter that uses the wrong calling mechanism causes CMS to fail.
- ID:149 - RESERVE/NOOUTPUT of a generation does not require RWD access to the element data file.
- ID:153 - Spurious characters included in WORD or LONGWORD formatted differences.

The output of the DIFFERENCES command formatted as WORD or LONGWORD includes spurious characters at the end of records. Characters are added to round up the record length to a multiple of the data partition size. These characters are repeated from earlier in the record.

- ID:154 - Incorrect new_element value in callback from CMS$SHOW_ GENERATION.

  CMS$SHOW_GENERATION, when applied to a library search list, sets the **new_element** argument of the callback to the value 1 for the first call for a new library in the list. The *DIGITAL Code Management System Callable Routines Reference Manual* states that the value 1 should only be used for the first call of the output routine.

- ID:165 - Long CMS messages are truncated.

  CMS uses the OpenVMS SYS$PUTMSG routine while processing messages to be displayed. SYS$PUTMSG is hard coded to limit message size to 256 bytes. With the addition of long variants, CMS messages can be much longer than that and the SYS$PUTMSG limitation can cause both Command Line Interface and DECwindows messages to be truncated.

- ID:166 - Some legal forms of extended file names are not allowed by create element and insert element. File names combining spaces and commas in unusual combinations are sometimes not allowed in an element name.

  A work-around is to use a wildcard specification to identify the file:

```
CMS> create element a^_^,.^, ""
%CMS-E-NOCREATE, error creating element a^_^,.^,
-CMS-E-ERRPAREXP, error parsing element expression
-CMS-E-NEEDPERIOD, element name must include or
     end with a period

CMS> create element  ^_.^, ""
%CMS-E-NOCREATE, error creating element DATA:[TST]^_.^
-RMS-E-SYN, file specification syntax error

CMS> create element ^_^,.^, ""
%CMS-E-NOCREATE, error creating element ^_^,.^,
-CMS-E-ERRPAREXP, error parsing element expression
-CMS-E-NEEDPERIOD, element name must include or
     end with a period

CMS> create element /keep *.^, ""
%CMS-I-CREATED, element DATA:[TST]^_^,.^, created
%CMS-I-CREATED, element DATA:[TST]^_.^, created
%CMS-I-CREATED, element DATA:[TST]a^_^,.^, created
%CMS-I-CREATES, 3 creations completed
```

- ID:170 - When the /LIKE qualifier is used with SET ACL, the library search list is not used to search for the source object. SET ACL/LIKE requires that both source and destination objects be in the same library.
- ID:171 - The object field in the history record is truncated to about 237 characters.
- ID:172 - When using extended file name specifications, an element with a name longer than 224 characters may be deleted completely from a CMS library if an attempt is made to MODIFY the element.

```
%CMS-S-MODIFIED, element
 DATA:[LIB1]1234567890A14A1234567890A28A1234567890A
 42A1234567890A56A1234567890A70A1234567890A84A12345
 67890A98A1234567890A112A.
 1234567890A127A1234567890A141A1234567890A
 155A1234567890A169A
 1234567890A183A12345a200a123456789
-RMS-E-CRE, ACP file create failed
-NONAME-W-NOMSG, Message number 00000000
-RMS-E-CRE, ACP file create failed
-SYSTEM-W-BADFILENAME, bad file name syntax
```

  This problem can also cause a REPLACE operation to fail with element name longer than 224 characters.

  The offending element name can be much shorter if it contains escaped sequences. In this case the escapes also count in the length of the name.

  Also, if you do CREATE element using an element name that is more than 255 characters after the escapes are added the error message says it is an illegal element name.

  A workaround is to use element names of 230 characters or less.
- ID:173 - Callable routine CMS$SHOW_RESERVATIONS sometimes reports a concurrent reservation as not concurrent in the **concurrent** value of the callback data when one specific reservation is specified for the **element_expression** on the call to CMS$SHOW_RESERVATIONS.

  A workaround is to use wildcards in the **element_expression**.
- ID:174 - Creating a CMS library with very long directory specification produces garbage message.

```
$ show default

DECSET6:[MWK.12345678901234567890123456789.
12345678901234567890123456789.
12345678901234567890123456789.
12345678901234567890123456789.
12345678901234567890123456789]

$ cms create libibrary [.12345678901234567890123456789] ""
  %CMS-I-FAOMISMATCH, FAO count of 1 does not match expected
   value of 2
  -CMS-I-TEXT, !AS is too long; maximum is !UL character!%S
  %CMS-E-NOCREATE, error creating CMS library
  -CMS-E-TOOLONG, directory spec is too long;
   maximum is 1769264 characters
  %CMS-W-UNDEFLIB, library is undefined
```

4.2 DECwindows Problems

The following known problems exist for the DECwindows interface in CMS Version 4.1:

- ID:19 - The DECwindows interface ignores accelerator keys.

  The DECwindows accelerator keys Ctrl/O (open), Ctrl/C (close view) and Ctrl/E (exit) have no affect in the DECwindows interface.

- ID:35 - DECwindows `CREATE ELEMENT` claims work in progress when it is not.

  Under certain circumstances, the CMS DECwindows `CREATE ELEMENT` function will pop up an erroneous work-in-progress box. It can be ignored.

- ID:76 - DECwindows and file specifications greater than 255 characters.

  The DECwindows interface may produce strange results if the user inputs a file specification that is greater than 255 characters.

  DECwindows does not check the size and blindly passes this on to the low-level routines, which blindly moves the size into a byte field before processing. Thus, a 256-character file specification, is exactly like providing none.

- ID:92 - Can not deselect REPAIR button in VERIFY dialog.

  Once the REPAIR button has been selected in the VERIFY dialog box from the MAINTENANCE menu, it can not be deselected.

- ID:100 - When VIEW RESERVATIONS is selected, and the user double-clicks on a particular reservation then selects CONCURRENT REPLACEMENTS, VIEW, EXPAND, ALL, an incorrect error message is reported.

- ID:105 - RESTRICT ELEMENT dialog box is laid out incorrectly.

RESTRICT ELEMENT displays a button on top of a text field.

- ID:107 - In the case of virtual memory being exhausted when updating the contents of a CMS View, if the view is iconified, an access violation will occur when CMS attempts to bring up a caution box. If the view is not iconified, after acknowledging the box, the cursor will remain in a ''watch'' shape and attempting to bring up a dialog box can result in an access violation.

  The problem can be avoided through the use of the Customize Restrict functionality. For example, by default `RESTRICT HISTORY` will display only the previous 30 days' history.

- ID:108 - If message logging has been customized to None, the Cancel Operation button may not be recognized until after the operation has been completed.

- ID:109 - Some informational messages resulting from opening or closing libraries will appear on the terminal where the CMS DECwindows interface was invoked rather than in the output window. This also occurs when manipulating a reserved element or an element under review. This is caused by routine CMS$DW_OUTPUT( ) not using the Message Logging Redirection Pointer.

- ID:164 - The DECwindows interface does not display all of the messages that the Command Line Interface does because the DECwindows interface uses SYS$GETMSG and SYS$FAOL instead of SYS$PUTMSG and calls the routines only once thus displaying only the first message in the signal vector.

4.3 Documentation Problems

- ID:111 - Using CMS$INSERT_GENERATION or CMS$INSERT_GROUP and the if_absent parameter set to true with a generation or group name that already exists in the class or group returns an error. The *DIGITAL Code Management System Callable Routines Reference Manual* states that the return is CMS$NORMAL.

- ID:112 - The /SUPERSEDE qualifier is negatable. Its description is missing from HELP.

- ID:116 - In the *DIGITAL Code Management System Reference Manual*, the description of the DIFFERENCES command should note that the /PARALLEL qualifier can only be used with record partitioning.

- ID:117 - The values for the access parameter of the CMS$SHOW_ RESERVATIONS callable routine are incorrect in the *DIGITAL Code Management System Callable Routines Reference Manual*. The correct values are:

  0 - Concurrent reservations are allowed.

  1 - Concurrent reservations are not allowed.

  3 - Existing reservation does not allow other reservations

- ID:124 - The remark parameter for the CMS$REMARK routine cannot be greater than 1000 characters.

- ID:130 - In the description of the CMS$REPLACE routine, the *DIGITAL Code Management System Callable Routines Reference Manual* states that the **confirm_routine** is used in CMS$SET_LIBRARY. The **confirm_ routine** is also available in CMS$CREATE_LIBRARY. In addition, the **confirm_routine** can be used for CMS$UNRESERVE, CMS$FETCH and the delete routines.

- ID:134 - HELP on DIFFERENCES/PARALLEL states that lines are truncated using a plus sign (+) when, in fact a asterisk (*) is used.

- ID:136 - Section 1.4.2 of the *DIGITAL Code Management System Callable Routines Reference Manual* incorrectly states that the only user modifiable part of the LDB is the fifth longword. In fact, the sixth longword is also user modifiable. The sixth longword contain the occlusion mask.

- ID:156 - The *DIGITAL Code Management System Callable Routines Reference Manual* incorrectly states that CMS$CREATE_CLASS and CMS$CREATE_GROUP operations are restricted to the first library in a search list.

- ID:168 - The *DIGITAL Code Management System Callable Routines Reference Manual* needs to better document the CMS$SHOW_VERSION routine including more examples and the following table:

  When the **absolute** argument is passed to callable interface routine CMS$SHOW_VERSION, the following absolute version numbers are returned for the CMS SSB releases:

```
V3.7    100177
V3.8    100184
V3.9    100189
V4.0    100191
V4.1    100194
```

- ID:175 - Section 3.1.1 of the *Guide to DIGITAL Code Management System for OpenVMS Systems* incorrectly states that OpenVMS systems limit directory trees to a depth of eight. This restriction is imposed by CMS, not OpenVMS.

5
Restrictions

The following restrictions apply to CMS Version 4.1:

- Rarely, when using CMS, the following error is encountered:

```
%CMS-F-BUG, there is something wrong with CMS ...
-CMS-F-NOMAP, Cannot map the appropriate control structure
-CMS-F-TEXT, dynamic buffer space exhausted. Reenter CMS.
```

  This error is caused by a design limitation in CMS. Under extreme conditions, CMS does not have enough buffer space allocated to complete an operation, and it cannot deallocate the exhausted buffer and reallocate it again. This error should only occur when using the `COPY ELEMENT` command, and only for attempts to copy elements with hundreds of generations (typically more than 500-600). This problem has nothing to do with the number of elements being copied, it only depends on the number of generations within any particular element.

  If the problem is encountered, the primary workaround is to reduce the number of generations in the element that caused the error. Typically, elements with this many generations have been automatically created or altered, and in many cases, the generation history isn't uniformly interesting. Judicious use of `DELETE GENERATION` could get the number of generations down to a manageable number.

- When using CMS library reference copies, certain file system related messages, such as RMS-F-DEV or RMS-E-PRV, may refer to the reference copy directory. The CMS messages do not make this explicit in some cases. When such problems are encountered, check the protections on reference copy directory.

- Using `VERIFY/RECOVER` on a library that does not need recovery, fails with an error diagnostic, RECNOTNEC, stating that the library does not need recovery:

```
$ CMS VERIFY/RECOVER
%CMS-E-NORECOVER, error recovering library
-CMS-E-RECNOTNEC, recovery is not necessary;
  the library is in a safe state
```

- Use of CMS/LSE integration features.

  To use the CMS/LSE integration features, introduced in CMS Version 3.8, LSE Version 4.3 is required. If an earlier version of LSE is used, the CMS/LSE integration features will not be available. No other features are effected by using an earlier version of LSE (prior to Version 4.3).

- The following restrictions apply to the `CMS DIFFERENCES` command when used with a CMS library search list.

  If both file_1 and file_2 are element generation specifications, then both of the elements must reside in the same library of the library search list.

  `CMS DIFFERENCES` will emit CMS-E-NOFETCH (or other messages returned by CMS$FETCH_OPEN) if it cannot locate a specified element generation. In a library search list, this may occur if one or both of the specified elements do not exist in the first library. These messages will be issued for each library of the library search list in which the element is not found. The exact sequence of such messages depends on the values of the `/OCCLUDE` qualifier. In such cases, a CMS-E-NOFETCH message may be followed by a CMS-S-IDENTICAL or CMS-W-DIFFERENT message. These messages indicate the actual status of the operation, whereas the intervening CMS-E-NOFETCH messages are a side-effect of the library search list traversal.

  The same considerations apply to the callable routine CMS$DIFFERENCES. CMS$DIFFERENCES uses the callable fetch routines CMS$FETCH_OPEN, CMS$FETCH_GET, and CMS$FETCH_CLOSE. Conditions may be signaled from these routines during the operation of CMS$DIFFERENCES.

- The DECwindows Options Save_Options feature overwrites the current CMS resource data file in the DECW$USER_DEFAULTS directory. If a version of the CMS resource data has been placed in this directory for customizing the interface for eXcursion small screen support, it will be overwritten if this option is used. The customized values will not be copied into the new version of the resource data file and will no longer have any effect on the CMS DECwindows interface.

- There have been several reports that CMS is corrupting source files via the use of the `/NOTES` qualifier.

This occurs because CMS is strictly syntax driven when /NOTES is involved. During a CMS Replace, it will look at any information contained at the end of each line and remove this information if it looks like a valid notes field. (A notes field is parsed as a ''left part'' followed by a generation number followed by a ''right part''.) A notes field NOT containing any ''left part'' or ''right part'' contextual information and containing only a generation number field (most examples supplied with the reports specified that they used the /NOTES="#G" form) is especially vulnerable since the format of the generation number is, for many of its valid forms, exactly the same format as a numeric constant which commonly may show up at the end of a line in most programming languages.

CMS does not (and probably cannot, in general) keep track of which specific lines it placed notes on. Thus when CMS replaces a file back into an element, it scans all input lines for notes information to be removed. The problem of remembering which lines had notes placed on them seems simple until you remember that the files have probably been edited so that the original context is relatively hard to recover except for the simplest cases. Even trying to depend on the /POSITION qualifier value to determine the ''validity'' of a particular string as a note is not trustworthy.

To avoid this problem, take full advantage of the complexity allowed in the notes string syntax and define the notes string to have a form that would be very unlikely to be encountered using your normal coding styles. For example:

```
/NOTES="/*--#G--*/"
```

This syntax would probably be acceptable to a C programmer whose coding style did not typically include the ''–'' prefix and ''–'' suffix in comments.

A side note: One of the COBOL examples received demonstrates this problem in a more subtle fashion. Specifically, when an element is reserved using /NONOTES, a subsequent REPLACE does not remember that it was reserved using /NONOTES and thus tries to remove the notes it finds (thus corrupting the file for the example received). The unfortunate part of this problem is that if we modify REPLACE to honor the state of the notes qualifier as reserved and NOT try to remove the notes from the file for this case, we now have a time bomb. The next time a RESERVE and REPLACE of the element is done WITH notes, the information will then likely be stripped off.

In summary, when using /NOTES be careful when picking the notes prototype string to avoid colliding with otherwise legal syntax in the language being used.

- CMS does not perform correctly when run from a detached process where no CLI present. Errors like LIB-F-NOCLI will occur.
- Callable CMS allows a command string to contain a maximum of 256 characters, where interactive CMS allows a maximum of 1024 characters. The workaround is to provide an input routine and use the continuation line when the library is set up.
- The history line sent to the CMS event handler and seen in notification messages will contain the letter 'W' after the transaction time field if the transaction resulted from a wildcard operation. This information should be suppressed, as it is by the `SHOW HISTORY` command.
- CMS Version 2.n attempts to operate on a CMS Version 3.n library, despite having incompatible formats. This may result in a corrupted CMS Version 3.n library. Note that this problem does not exist for libraries created by or converted to CMS Version 3.5-5 or later.
- A maximum of 65,536 characters can be copied from a CMS view to the DECwindows Clipboard.
- When using CMS with extended filenames from the DCL command line, the DCL parse style must be set to extended filenames. Use the DCL command `SET PROCESS/PARSE_STYLE=EXTENDED`. For example:

```
$ SET PROCESS/PARSE_STYLE=EXTENDED
$ CMS SET LIBRARY DISK1:[PROGRAMS.CMSLIB_EXT]
$ CMS DIFFERENCES Parser^[Old_Version^].CPP
  Parser^[New_Version^].CPP
```

- If an attempt is made to use an earlier version of CMS (prior to Version 4.1) for access to a library that has the long variant name attribute set, an error message will be displayed. This will occur for any command issued against the library by an older version of CMS until a CMS Version 4.1 user successfully restores the /NOLONG_VARIANT_NAMES attribute on the library. This is the exact error displayed for this condition:

```
%CMS-F-BADLIB, there is something wrong with your library
* CMS-F-BADTYPSTR, header block type is 145; it should be 17
```

- While using CMS Version 4.1 to VERIFY a library that was created by an earlier version of CMS (prior to Version 4.1), errors such as the following may be displayed because the new commands that were added in Version 4.1 have not yet been added to the library control file:

```
%CMS-E-NOTFOUND, command DISK1:[CMS.LIB1]COPY_CLASS not found
-CMS-E-USEREPAIR, use VERIFY/REPAIR
%CMS-E-NOTFOUND, command DISK1:[CMS.LIB1]COPY_GROUP not found
-CMS-E-USEREPAIR, use VERIFY/REPAIR
%CMS-E-NOTFOUND, command DISK1:[CMS.LIB1]MODIFY_RESERVATION
   not found
-CMS-E-USEREPAIR, use VERIFY/REPAIR
%CMS-E-ERRVERCON, control file verified with errors
```

Use VERIFY/REPAIR, if it is unable to correct the problem, use VERIFY /REPAIR a second time to complete the repair of the library and add the new commands to the library control file.

- CMS still restricts directory trees to maximum depth of eight, even though OpenVMS no longer imposes that restriction.

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

In addition, Compaq 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://ww1.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 Support Center host system.

Customers can register for DIA and receive a WIS account by contacting their Customer Support 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 request, please include the following information:

1. CMS version number 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 diskette, or tape cassette.