CMVC frequently asked questions: miscellaneous topics (users, defects/features, access, etc.)

Document Number TR 29.2329

Angel Rivera, Lee Perlov

CMVC/TeamConnection Development
IBM Software Solutions
Research Triangle Park, North Carolina

ABSTRACT

This technical report provides answers to frequently asked questions made by CMVC users with respect to miscellaneous topics:

- End user tasks
- Defects and features
- Administration of users
- Components and authority access
- Special reports
- Notification
- Miscellaneous

ITIRC KEYWORDS

- CMVC
- components
- defects
- features
- users
- access
- notification
- reports

ABOUT THE AUTHORS

ANGEL RIVERA

Mr. Rivera is an Advisory Software Engineer and team lead for the development of CMVC 2.3. He joined IBM in 1989 and since then has worked in the development and support of library systems.

Mr. Rivera has an M.S. in Electrical Engineering from The University of Texas at Austin, and B.S. in Electronic Systems Engineering from the Instituto Tecnológico y de Estudios Superiores de Monterrey, México.

LEE R. PERLOV

Mr. Perlov is a Staff Software Engineer in the TeamConnection/CMVC development group. He started working for IBM in 1985 in Gaithersburg, Md, working in the Federal Systems Division on various projects for the United States intelligence community. He then moved to RTP to work on library development and support.

Mr. Perlov received a B.S.Acc degree in Accounting from the University of Florida in 1983. He also completed two years of graduate work in the Department of Computer Science at the University of Florida.

CONTENTS

FIGURES

INTRODUCTION

This technical report provides answers to frequently asked questions made by CMVC users with respect to miscellaneous topics (users, defects/features, access)

This technical report covers the following topics:

- General
- End user tasks
- Defects and features
- Administration of users
- Components and authority access
- Special reports
- Notification

ACKNOWLEDGEMENTS

Many of the questions and answers that are compiled in this technical report were obtained from the CMVC forum in the IBMPC conferencing disk, and from the CMVC6000 forum in the IBMUNIX conferencing disk.

We want to thank the main participants in these electronic forums for their support!

Also, we want to thank Dodde Stark for editing this technical report.

GENERAL

BRIEF HISTORY OF DELIVERABLES FOR CMVC 2.3

This is a brief history of the deliverables for CMVC 2.3:

1. CMVC 2.3.0.0: December 30, 1994

   The GA for CMVC 2.3 (2.3.0.0) was in a standalone CD-ROM The actual level from our CMVC family was 2.3.0.12, but this fact was not visible to the users.

2. CMVC 2.3.0.14: June 30, 1995

   The next public delivery of CMVC was with another standalone CD-ROM called version 2.3.0.14. From this moment on, we started using the 3rd number to indicate the modification and the 4th number to indicate the level. This was the last time that we issued a standalone CD-ROM for CMVC.

3. CMVC 2.3.0.18: December 1, 1995

   The next public delivery of CMVC was with level 2.3.0.18 and we participated with the AIX Version 3 Software Solutions Showcase CD-ROMs.

4. CMVC 2.3.0.22: June 11, 1996

   The next public delivery of CMVC was with level 2.3.0.22 and we participated with the AIX Version 4 Software Solutions Showcase CD-ROMs, 5th edition.

5. CMVC 2.3.0.25: June 30, 1997

   In May 1997, from our level 2.3.0.24 (version 2.3.0 uses 2 digits to represent the year) we branched to another release, called 2.3.1 which uses 4 digits to represent the year and which is Year 2000 compliant.

   Then we proceeded to release 2.3.0.25 (2 digits for the year) to the public through the external ftp site. We need to keep working in this release because there are old platforms and databases that we still need to support and which we do not think is worthy to support for CMVC 2.3.1, such as AIX 3, DB2 V1, Informix 5, SunOS 4.1, HP-UX 9.

6. CMVC 2.3.1.1: June 30, 1997

   Finally, at the end of June we are releasing 2.3.1.1 (4 digits for the year) to the external ftp site; only the latest platforms and databases are handled: AIX 4, DB2 V2, HP-UX 10, Informix 7, Solaris 2.5, Oracle 7.1/7.2 (we cannot run on Oracle 7.3). Not all the combinations are ready yet, but we are working on them.

IS CMVC GOING TO WORK WITH THE YEAR 2000?

Question:

Is CMVC going to work with the Year 2000?

Answer:

CMVC 1.x and 2.3.0.x store dates with only two digits for the year. Thus, the sorting of objects by date may give incorrect results, because the year "1997" is represented as "97" but the year "2000" is represented as "00". CMVC does not manipulate in any way the year information and relies on the underline database for the sorting.

In mid-1997, the release 2.3.1.1 was branched from 2.3.0.25 in order to use 4 digits to represent the year, and thus, 2.3.1.x is Year 2000 Compliant.

DOES CMVC SUPPORT ENCRYPTION BETWEEN THE CLIENT AND THE SERVER?

Question:

Does CMVC support encryption between the client and the server?

Answer:

CMVC does not explicitly support any encrypting mechanism.

However, as long as the CMVC server and CMVC client are not aware of any data encryption, it would be possible to do encryption (probably by hardware) in the communications channel.

DOES CMVC COMPLY WITH MILITARY SPECIFICATIONS?

Question:

Does CMVC comply with military specifications, such as MIL-2167 and MIL-0937?

Answer:

CMVC does not have to conform to military specifications, but the organization using CMVC needs to do it. In order for the organization to comply with these specifications, the organization must collect certain information and follow a certain process.

Through its configurable fields, user exits, selectable processes and e-mail notification, CMVC can support an organization that needs to operate according to military standards. In installations like this, you most certainly need a consultant that knows how to configure CMVC to use the labels and definitions from the military specifications. Also, the development organization of the customer usually decides to what extent they must follow the military specifications and which processes to use during various development phases. Here, the configurability of the software CM tool is essential, as it needs to support their process and collect the information they need.

ARE CMVC AND TEAMCONNECTION COMPATIBLE BETWEEN EACH OTHER?

Question:

Are CMVC and TeamConnection compatible between each other?

Answer:

Although TeamConnection is the successor product to CMVC and there is a lot of common functionality between these products, there is NO compatibility between CMVC and TeamConnection. That is, a CMVC client cannot work with a TeamConnection server, nor can a TeamConnection client work with a CMVC server.

TeamConnection provides migration tools to help migrate a CMVC family to TeamConnection.

For more information on CMVC/TeamConnection, please see the following technical reports:

```
29.2253  Comparison between CMVC 2.3 and TeamConnection 2

29.2254  Migrating CMVC 2.3 to TeamConnection 2
```

The big differences between TeamConnection and CMVC are shown in the next subsections, which were extracted from the TR 29.2253 mentioned above.

The big differences for users

Here are a few things that will be significant changes for CMVC users migrating to TeamConnection.

- The names of several objects have changed. For example, tracks are now workareas and levels are now drivers.
- The versioning scheme is different:

  - Files (parts) are NOT versioned by checkout/checkin, but by the number of freezes in a workarea.
  - Version numbers are named relative to the workarea (for example, the part a.c can have a version number work1:3).
  - Drivers and releases are essentially specialized workareas, and therefore versioned.
- Workareas are much more important than tracks and behave differently.
  - A workarea needs to be specified in all teamc part -checkin and -checkout commands. Even if you are just using TeamConnection to store parts in a single release without any defects or features, you need to create a workarea and periodically integrate and commit your changes.
  - You will periodically be required to refresh your workarea before you perform actions like integrating the workarea, or checking parts in or out. For example, if you try to check out a file that another user has updated in a different workarea, but has not committed, you will need to refresh from that workarea.
  - You must freeze a workarea (explicitly or implicitly) to create new versions of parts changed in that workarea.
- When performing queries, like filling in a filter window, you are often querying within a context instead of the entire family. You need to specify that context by entering release, driver or workarea. If you do not provide enough context you will get an error or not be able to press the OK button.

The big differences for family administrators

Here are a few things that will be significant changes for CMVC Family Administrators migrating to TeamConnection.

- There is a single installation for the TeamConnection software and the ObjectStore database (which is used by TeamConnection). You do not need to be an expert at a database before administering TeamConnection families.

  Note: Object Design Inc. provides courses on the administration of ObjectStore.
- All of the data is stored in the database, including files. You will not need to know SCCS in order to keep the database and file versions synchronized. This also makes backup much easier.
- Most of the administration commands have changed. See the **TeamConnection Administrator's Guide** for more details. Further, a separate GUI is provided for family administration on the Intel operating systems. It is planned to port the GUI to the Unix operating systems.

- TeamConnection build administration requires maintaining pools of build machines so that TeamConnection can properly perform release builds. This means that TeamConnection requires more extensive planning and management for the network.
- The migration facility is different in TeamConnection than in CMVC. Further, we recommend you follow the process documented in the Technical Report, **Migration from CMVC 2.3 to TeamConnection 2.0**.

The big differences for tools integrators

CMVC relied on framework tools such as **SDE/Workbench** to provide tool integration. SDE/Workbench provided build integration through "make", GUI integration and messaging, etc.

TeamConnection is actually a point of integration, and it provides:

- A generic build interface that allows for traditional builds, as well as packaging and distribution.
- A client cache that allows client applications to extract and update data in the TeamConnection family without calling client commands.
- An API to allow client applications to perform TeamConnection tasks using function calls instead of client line commands.

WHAT ARE THE KNOWN INCOMPATIBILITIES WITHIN THE CMVC VERSIONS?

Question:

What are the known incompatibilities within the CMVC versions?

Answer:

CMVC version 2 is not compatible with CMVC Version 1 nor with any version of TeamConnection.

Any CMVC V2 client can work with any CMVC V2 server, the exceptions are the OS/2 and Windows client that require at least a CMVC 2.2 server.

There are no other restrictions with respect to operating systems. For example, a CMVC V2.1 client in HP-UX can work with a CMVC V2.3 server on AIX.

CLARIFICATION OF SIZES IN CHAPTERS 6 AND 7, CMVC USER'S REFERENCE

Question:

Clarification of sizes in Chapters 6 and 7, CMVC User's Reference.

Answer:

The sizes for most of the fields in the Chapter 6 (Views for Queries) and Chapter 7 (Tables for Queries) of the CMVC User's Reference manual show a number that is higher than the one actually used by the CMVC server which works with a number of characters that is half that original number.

This means that if Release Name has a size of 31 (characters) it can only accept 15 bytes.

This method forces a limitation of Single Byte Character Set (SBCS) and Double Byte Character Set (DBCS) strings to the number of characters. This means that although the Release Name has the field length of 31 bytes, it can accommodate only 15 SBCS characters and 7 DBCS characters.

Note: The reason for a 31 bytes is that the field allows 2 bytes for each character, plus 1 byte for the NULL to terminate the string.

USING DIFFERENT TIME ZONES WITH THE CMVC CLIENT AND SERVER

Question:

Without actually changing the time zone used on the host, is there any way to convert dates by time zones within CMVC?

Answer:

The Date and Time is a field obtained from the database (it is a string), and it is the date and time of the CMVC server.

There are no functions in CMVC that will change the value of this string to accommodate for the different time zones between the server and the client.

Other tricks with time

There are several other cases where it is useful to change the system time temporarily in order to do something in CMVC. We do not strongly recommend these, but some of our customers have found these steps useful.

- Checking out a file that has not been changed in over a year.

  SCCS returns a warning when checking out a file that has not been changed in over a year. This is distracting for CMVC. If you have lots of such files to change, it is better to backdate your system while checking out, then set the system back to current time to check in.

- Restoring from an archive.

  If you need to restore from an old archive, that is, create a new family from an archive. Rolling back the date of the system will give a consistent date for all restored objects, including the database.

WHAT IS THE AUTHENTICATION MECHANISM FOR CMVC?

Question:

What is the authentication mechanism for CMVC? How could I become another CMVC user?

Answer:

The authentication mechanism for CMVC is as follows:

- You need to be authorized to access CMVC from a known host and as a known CMVC user id. For example, in the case of user Angel Rivera that logs in as "rivera" to the host "cordoba" (a UNIX host) and wants to interact with CMVC with the CMVC userid "rivera".

- However, if we rely only on those 2 variables, then anybody that has access to the host "cordoba" could login into a CMVC family as a superuser, and possibly cause havoc in the family.

- Therefore, a third variable comes into play: because CMVC does not have passwords for users, but the UNIX operating system does, then CMVC forces to you login into the operating system (and thus, you have to provide a password for authentication to the operating system).

  When you do this login action, you have to use a name; in this example is the system login "rivera". We called it the system login id, and this is the 3rd variable that is needed.

  Because the OS/2 and Windows operating systems do not have a login mechanism, the CMVC_USER and USER variables are used as the system login.

- So you need to have an authorized set of these 3 variables registered with CMVC in order to be authenticated as a valid user.

  This scheme allows to have multiple roles with CMVC. Continuing with the example, the user has both a CMVC normal user id "rivera" and a CMVC superuser id "su_angel", but only one system login: "rivera". Notice that this user does not have a special UNIX account called "su_angel". Thus, the user has 2 hosts lists, both of which have the same system login (rivera) and host, and one host list has the CMVC user id of "rivera" and the other has the user id of "su_angel"

- Unfortunately, CMVC is NOT consistent with the usage of Login and Userid. Sometimes they are mixed up, and this causes confusion, specially with the line commands, which were created first in the history of the evolution of CMVC. Fortunately, the GUI tries to alleviate this problem and it is more consistent with the terms.

WHY A LONG FAMILY NAME (WITH PORT NUMBER) HAS PRIORITY OVER SHORT NAME?

Question:

Why a long family name (such as familyName@hostName@4321) has priority over short name (such as familyName or familyName@hostName?

Answer:

- The long family name format is:

  ```
  familyName@hostName@portNumber
  ```

  When you specify the hostName and port number, CMVC does not do any name resolution and thus, this format has precedence over the short family name (see next paragraph). Thus, you can put anything you want as the "familyName" and you will still connect to the specified port on the specified host.

- The short family name format is:

  ```
  familyName
  ```

  When you specify just the family name without the @'s, CMVC uses the name to find the host name and port number from the nameserver (or the local /etc/hosts file) and the /etc/services file.

CLARIFICATION OF FLAG ARGUMENTS, IN PAGE 3 OF COMMANDS REFERENCE

Question:

Clarification of Flag Arguments, in page 3 of Commands Reference

Answer:

The "Name" value that follows a flag of a CMVC command has another restriction besides the ones mentioned in the table in page 3 of the CMVC Commands Reference:

The Name cannot begin with a + (plus) or a - (minus) character. This limitation is needed to avoid confusion with other Flags, which always begin with any of these 2 characters.

END USER TASKS

SAMPLE PROFILE FOR A CMVC CLIENT

Question:

Is there a sample profile ready-to-use for a CMVC client?

Answer:

Starting with CMVC 2.3.0.18, we are providing an excellent sample profile for a CMVC client. We strongly recommend that every client, regardless of the expertise of the user, should use the CMVC client sample profile as the profile for the CMVC client (or at least to be used as the basis for the profile). We are constantly adding new environment variables and adding new hints and tips.

The client sample profile has variables that are dependent on the operating system and which are commented out; it is the responsibility of the CMVC user to review the entire profile and uncomment those variables that are appropriate.

The CMVC client sample profile is located in:

```
/usr/lpp/cmvc/samples/profile.user
```

HOW TO GET THE FIRST BASELINE IN CMVC

Question:

Now that CMVC is installed and ready to be used, what should I do to get the first baseline of the files that I have in my workstation and that I want to store in CMVC?

Answer:

If you are a novice user, it is recommended to use the CMVC GUI for now. This example will emphasize it. If you are using the CMVC GUI for OS/2 or Windows, then you will find everything you need in the "selected" menu. In UNIX, use the "actions" and "show" menus.

Here are the steps to follow to obtain your first baseline of the files to be stored in CMVC by using tracks and levels, which are strongly recommended for normal production work.

1. The CMVC family administrator or a superuser needs to create the following:

   - Users and their host list entries.
   - The component hierarchy.
   - The release or releases.
   - The authority for the users to access the component hierarchy.

2. Ensure that the process configuration for the main component where you will open the initial feature has the following subprocesses:

   - dsrFeature is probably on but not necessary for you right now

3. Ensure that the process configuration for your release has the following subprocesses:

   - "track" should be "yes".
   - "approval" is probably "yes" but is not necessary for you right now.
   - "fix" should be "yes".
   - "level" should be "yes".
   - "test" is probably "yes" but is not necessary for you right now.

4. To exploit the values from the Options (UNIX) or Settings (OS/2 and Windows) window, you could setup the following default values:

   For UNIX, OS/2 and Windows:

   - Relative directory:

     Specify the full path name of the directory that is the top one that has the beginning of your file structure. For example, let's suppose that you have the following files in your workstation and you want to create them in CMVC:

     ```
     /home/user1/src/cat/file1
     /home/user1/src/file2
     /home/user1/makefile
     ```

     Later on when you extract the files from CMVC you want to preserve the original meaningful directory path:

     ```
     src/cat/file1
     src/file2
     makefile
     ```

     In this case you need to specify that the relative directory is:

     ```
     /home/user1
     ```

   For OS/2 and Windows only:

   a. Component.

   b. Release.

5. Bring up the Features window.

From the pull down menu, create a feature which will be used to create the files into CMVC from the files in your workstation to obtain your first baseline.

6. Refresh the list of the Features window in order to see the entry for the new feature.

7. If dsrFeature is on, you need to perform the "design, size, and review" steps, by highlighting the feature and selecting the proper action from the pull down menu.

   You need to provide a value for the remarks when doing the design step. However, for the other steps, you may not need to specify remarks.

8. Accept the feature. This action moves the feature to the Working state.

9. Create a track for your feature and the desired release.

   If the approval subprocess is active for the release, then the track is created in the Accept state. Otherwise, the track is created in the Fix state (we are assuming that this subprocess is active for this example).

10. From the Features window, highlight the desired feature and select the action Show Tracks. This will bring up the Tracks window.

11. From the Tracks window then you could select to show the Approval Records (if using the approval subprocess) or the Fix Records.

12. If the approval subprocess is on, then you need to show the Approval records for your track and accept them; you could also abstain.

    Any of the above actions will move the track to the Fix state.

    However, if you reject the approval records, then the track does not move to the Fix state.

13. Once a track is in the Fix state, then you can now handle files.

    From the CMVC Task window you could bring up the Files window and select the Create action from the pulldown menu.

    You could use the Choices button next to the path name field. This will show you all the files that are available from the relative directory specified in the settings.

    Note: When using OS/2 and Windows, the file names are shown IN UPPERCASE (they are provided in this format by the Operating System, though). However, if you want to develop in multiple platforms such as UNIX (which is case sensitive), then we recommend that the files should be created in lowercase.

    If you care about the permissions of your files, then create at the same time those files that have the same permissions.

    When handling a list that has multiple files in the same action, CMVC takes the file permissions from the first file and uses them for the rest of the files in the list.

14. Because each file in CMVC is associated with a component, then whenever a file change is done (create, checkin, delete, rename) then a fix record is created for each affected component.

    The idea behind this concept is that the component owner should be aware of the file changes and approve/abstain/reject them.

15. After you have finished creating and checking in the files from your workstation, then you are ready to complete the fix records.

    From the Track window, highlight the desired track and select the action that shows the Fix records.

    The owner for each component involved in the track should handle the fix record and complete it.

    This actions moves the track to the Integrate state.

    Note: You need to have level process on at this point, or the track will move all the way to complete.

16. From the CMVC Tasks window select the Levels window. Then from the Levels window create a new level for the desired release.

    A level is a very useful object in CMVC and is used to gather tracks (and the associated file changes) that need to be built together.

17. Create a level member for the track.

    One way to accomplish this action is by highlighting the desired level and then performing the action Add Level Member.

18. Perform a "delta" level extract to a new directory in order to verify that everything is checked in correctly.

19. If you discover that a file is missing, then do the following:

    a. Delete the level member.
    b. Change the track to the fix state.
    c. Change the fix record to active.
    d. Go back to 13 on page 15

20. If all the files are now in the level, then commit and complete the level.

    That should give you a permanent baseline to start from. You can do later a full extract of this level in order to obtain this baseline.

USAGE OF NULL TO RESET 0 CHARACTERS IN CERTAIN ATTRIBUTES IN CMVC

Question:

Usage of NULL/null to reset 0 characters in certain attributes in CMVC.

Answer:

Currently it is possible to specify the NULL or null keyword to reset to null (that is, to 0 characters) certain attributes in CMVC, but this is possible only under the following circumstances:

- Only from the line command (it is not possible to do it from the GUI) and using the flag -modify.
- Only values that can have free-text can be changed to null, such as the reference field in defect/feature by using the keyword null (or NULL) such as:

  ```
  Defect -modify 247 -reference null
  ```

  The use of "null" in this instance will actually change the contents to zero characters, and will not use the word "null" in the reference field: that is, the field will be blank.

The exceptions to the use of keyword "null" to make null an attribute are:

- Cannot use for configurable fields, such as "target" in defect/feature.
- Cannot use for fields that expect a configurable value (that have a "choices" button in the GUI), such as "severity" in defect/feature.
- Cannot use for required fields, such as "abstract" in defect/feature.
- Cannot reset to null a free-text field that accepts blank spaces.

If you need to 'blank' an attribute, it is possible to use the line command with the -modify flag to reset the attribute to a single blank character by using " " as shown in the following example:

```
  Defect -modify 247 -reference " "
```

However, this procedure does not actually reset the contents of the attribute to 0 characters. Any searches that use this field, should look for " " instead of using the SQL operators 'is null' or 'is not null'.

IMBEDDING NEWLINE CHARACTERS INTO COMMENTS

The CMVC GUI makes it easy to imbed newline characters into remarks, but the command line leaves you to figure this out on your own. Here are some useful techniques:

Unix

Unix makes it very easy to imbed newline characters into a string:

- Escaping the newline:

  You can escape a newline character with a backslash, '/'. For example:

  ```
  Defect -open ... -remarks "This is line one \
  and this is line two"
  ```

- You can also redirect from standard input:

  You define an "end-of-file" character string (such as !EOF), then redirect from standard input. This uses a feature of the CMVC command line interface to support redirection of standard input. Here is an example:

  ```
  Defect -open ... -remarks - -family octo <<!EOF
  This is line one
  and this is line two
  !EOF
  ```

Quoting rules in Unix shell scripts

In the above examples, quotes (") were used. There are some important quoting rules to keep in mind when using Unix:

- No quotes allows the shell do anything it wants.

  There is a very long list of operations performed on the commands you type in a command shell. The most obvious are the ones listed below. In only a few cases is the order really important. If the two examples below are not sufficient, look at a book on Unix shell scripts.

- Quotes delay interpretation until after filenames are expanded.

  This is a subtlety, but in order for many commands to interpret a string like "*.c", it is necessary to avoid letting the shell convert the string to something like: a.c b.c c.c That is what quotes do for you.

  However, quotes do not prevent the shell from value substitution. For example, "$HOME" will be substituted for something like: /home/guest

  Quotes are usually what you want to use.

- Nothing that begins with a dollar sign ($) is interpreted in the current shell.

  A string like '$HOME' will print exactly as seen, without $HOME being replaced with the value for the environment variable. This is useful when you want to pass something to a shell script for interpretation unaltered. If you pass '$HOME' as an argument to a shell script, it will arrive as $HOME and be subject to interpretation there.

OS/2 and Windows

This should also work in the Windows operating system variations.

OS/2 is very stubborn when it comes to imbedded cr/lf characters, but it is possible.

- Escape character: ^

  In OS/2, the escape character is a caret, ^. In order to continue a remarks you must do the following:

  ```
  defect -open -remarks "this is a long"^
  More? "description of my problem"
  ```

  You will be prompted with `More?`, but the line will continue.

  You will find that the escape feature in OS/2 is more limited than the one in UNIX, but the basic capability is there.

- Redirecting standard input.
  - Just like Unix, you can use '-' as the value for remarks in a CMVC line command, then redirect the value from a file.

    ```
    Defect -open ... -remarks "This is line one ^
    and this is line two"
    ```

  - OS/2 has different syntax for redirecting from standard input. Start by redirecting from the console, "<con:", then conclude the input string with Ctrl-Z in the first column of a line. Here is an example:

    ```
    Defect -open ... -remarks - -family octo <con:
    This is line one
    and this is line two
    <Ctrl-Z>
    ```

HOW TO DELETE AN OBJECT IN CMVC

Question:

As a test, I created a component named test1, and then immediately deleted it.

I did a Show All Components query and the test1 component still shows up. How do I delete a component such that it will no longer be listed on a Show All Components query?

Answer:

When you "delete" most of the objects in CMVC, they are not really deleted, but instead, the Drop Date (or similar field) is filled in. This indicates to CMVC that the object is conceptually deleted. Then, if you want to recreate the "deleted" object later on, CMVC will just erase the value of the Drop Date.

If you do not want to see the "deleted" items, you will need to specify the following in the query:

```
and dropDate is null
```

For example, to list all the components, except the deleted ones, you can issue the following query from the Components window:

```
name like '%' and dropDate is null order by name
```

However, there are other objects for which the delete action means to be completely deleted from the database, because these objects do not have a dropDate field. These objects are the lists, such as host list, fix record list, etc.

CAN THE TEST RECORDS BE ASSIGNED TO SOMEONE OTHER THAN THE ENVIRONMENT PERSON?

Question:

We are using the test subprocess and we have a large number of testers. The test records are created with the environment person as the owner. We do not want to assign all the test records to only one person and then ask that person to reassign them to the other testers.

If we do not manually change the owner of the test records, is there a way for tester A to see what is their waiting workload (defects they opened which have unmarked test records)?

Answer:

One suggestion is to use the Notification Lists.

The CMVC family administrator could create a new notification group which will include the TrackTest action, and then the testers could subscribe to that notification group for the component in which the release was created.

Note: The only purpose for creating a release in a given component is to inherit the access lists and the notification lists for that component.

WHAT ARE THE PREREQUISITES FOR ENABLING/DISABLING SUBPROCESSES?

Question:

What are the prerequisites for enabling/disabling subprocesses?

Answer:

Because of the asynchronous relationship between the predefined CMVC process configuration and the CMVC process being used by releases and components, the family administrators should properly manage the update of these predefined CMVC process (especially for deleting and modifying existing predefined CMVC processes). If it is mismanaged, user confusion may result but the integrity of the CMVC configuration will not be impacted in any way.

The prerequisites for disabling a subprocess are shown below:

Figure 1. Prerequisites for disabling a subprocess.

Subprocess	Prerequisite
dsrDefect, dsrFeature	No defect or feature owned by the component is in design, size or review state.
verifyDefect, verifyFeature	No verification record owned by the component is in ready state.
approve	No track associated with the release is in the approve state.
fix	All fix records associated with the release are in complete state.
level	All levels are either in complete or working state, and, no track is in integrate state.
test	No test associated with the release is in ready state.

The prerequisites for enabling a subprocess are shown below:

Figure 2 (Page 1 of 2). Prerequisites for enabling a subprocess.

Subprocess	Prerequisite
dsrDefect, dsrFeature	None
verifyDefect, verifyFeature	None
approve	An approver list[4] must exist for the release.
fix	None

Figure 2 (Page 2 of 2). Prerequisites for enabling a subprocess.

Subprocess	Prerequisite
level	No track is in fix or integrate state.
test	An environment list[4] must exist for the release.

1. In general, the prerequisite is checked on a component when the process of the component is being modified. However, in case a defect or feature is assigned to a new component, the prerequisite is checked on a defect (or feature).

2. When the 'verify' subprocess is disabled and after the CMVC server successfully validates the prerequisites, the CMVC server will remove all the 'notReady' verification record associated with the component. If a defect (or feature) is re-assigned from a component with 'verify' subprocess to another component without this subprocess, then CMVC server will remove all the 'notReady' verification record for that defect (or feature).

   Those deleted verification records will not be recreated at the time the 'verify' subprocess is enabled or the defect (or feature) is re-assigned to a component with 'verify' subprocess, but are recreated at the time when the defect and/or feature are ready for verification.

3. When 'test' subprocess is disabled and after successfully validating the prerequisite, CMVC server will remove all the 'notReady' test record associated with the release.

   Those deleted test records will not be recreated at the time the 'test' subprocess is enabled but are recreated at the time when the track are ready for testing.

4. Flags are provided in the **Release -create** command to allow the user to create the approver list and environment list. Example:

   ```
   Release -create releaseName -process processName -environment envName
   -tester testerName -approver approverName
   ```

USING UNIX ALIAS AND WRAPPERING CMVC COMMANDS

For some CMVC users, a specialized client interface makes more sense than the current, very flexible and complex, command line interface. For many users, they need to interface with such a small part of CMVC (for example, only opening defects and processing verification records) that the GUI is also more than they need. These customers can do two relatively simple changes outside of CMVC:

1. Aliasing commands.

   Unix provides a simple aliasing tool for Korn shell. There is also one for C-Shell, but we will only demonstrate the Korn shell here. All the 'alias' Korn shell command does is to replace the first word typed by the user with a string that can be more than one word long. Here is an example:

```
alias NewDef="Defect -open -family ${CMVC_FAMILY} -sev 3 ... -remarks"
```

Now, when a user types `NewDef "A new problem ..."`, a defect is opened with the remarks specified, and everything else hardcoded or taken from environment variables.

2. Wrappering a command.

   This is a very simple concept, and if you need to do this you probably already have. However, if you are new to CMVC you might not have gotten around to it yet.

   Here are some reasons to wrapper a command:

   - Save typing, such as when certain parameters are always the same
   - When users need to "become" several different CMVC users, based on the task they are performing.
   - When starting the GUI or running commands for several different families.

   Since your users are already pointing to the cmvc/bin and cmvc/samples directories, there is no reason to not add shell scripts of your own to the samples directory. If you are worried about overwriting your tools when you do a CMVC upgrade, then put your tools in another directory and symbolically link the files into the samples directory. If you look closely at the CMVC installation, you will see we make use of symbolic links.

   As for the actual tools, just pick a name (such as NewDef), put the file into the samples directory and make it executable. By the way, we provide many samples with CMVC. The following is a simple but effective start:

   ```
   #!/usr/bin/ksh
   echo Executing $0:
   echo parameters: $*
   ```

DEFECTS AND FEATURES

CANNOT USE A BINARY FILE AS INPUT TO THE REMARKS FOR A DEFECT

Question:

While opening a defect, when entering remarks, there is an option to insert the contents of an existing file. However, when trying to insert a file that has binary characters in it, we get a partial file of the ascii up to the binary character. In other words the text is cut at the binary character. Does the remarks field take only ascii characters?

Answer:

Yes, only non-control characters are allowed in the remarks.

CANNOT MODIFY OLD REMARKS IN A DEFECT

Question:

I discovered a typo in the remarks that I used to open a defect. Now I would like to fix the typo, but I cannot modify these old remarks.

Answer:

The remarks are part of the audit trail for a defect or feature; this audit trail cannot be modified, it can only be appended, that is, you can only add new text to it, but you cannot delete or modify old text.

IS IT NORMAL TO HAVE DUPLICATE VERIFICATION RECORDS?

Question:

When showing verification records of a defect, we got multiple entries with the same defect number but different originators. Is this normal? What is the best way to close these defects?

Answer:

Assuming that other defects or features were returned as duplicates of the defect with the duplicate verification records, this is normal behavior.

Each owner of a verification record has to accept, reject or abstain on his/her record for the defect to get closed. This allows submitters of defects or features that were returned as duplicates to verify that the change they wanted actually got implemented.

HOW TO LIST ALL THE DEFECTS OF A COMPONENT AND ITS CHILDREN

Question:

How to do I list the defects of a component and its children?

Answer:

Anyplace where you can specify compName='xyz', you can substitute it with the following:

```
compid in
   (select childId from CompMemberView where parentCompName='xyz')
```

You will have to do this from the entry field at the bottom of the query window, but you can save the query to the task list or as a default query.

Note: Some databases are case sensitive. The correct case for each database field is shown in the manual **CMVC User's Reference**.

PROPOSED WORKAROUNDS FOR COMMON COMPLAINTS ABOUT DEFECTS AND FEATURES

- What to do when a verification record is rejected?

  One design problem of CMVC with respect to defects is that when a verification record is rejected, then the defect is moved to the state 'closed' instead of reopening the defect or opening a new defect.

  You can use the user exit cloneDefectSilent as explained in /usr/lpp/cmvc/samples/server.samples to clone the existing defect when the verification is rejected.

- A better way to use verification records

  Set authorities so that every user can accept/reject their own verification records, then abstain everyone else's verification records. In that way, only one person verifies a defect/feature. This could also benefit from a user exit.

- Showing relationship between defects and features during development.

  Use the reference field on defects to point to related features.

  Use naming conventions to show relationships. For example, if the feature is 1410, then the defects start with 1410.1 and continue by incrementing the last digit: 1410.2, 1410.3, etc.

ADMINISTRATION OF USERS

SUGGESTION THAT EVERY CMVC SUPERUSER SHOULD HAVE AT LEAST 2 USER IDS

Question:

Suggestion that every CMVC superuser should have at least 2 user ids.

Answer:

It is strongly recommended that every CMVC superuser in your family should have at least 2 CMVC user ids: one as a normal user and another as a superuser. Furthermore, only when doing CMVC tasks that require superuser authority should the CMVC superuser id be used; for all other CMVC activities, the non-superuser id should be used.

The reason for this recommendation is to avoid that, by accident, a CMVC superuser could bypass the normal safeguards of CMVC and cause problems or confusion with other users. For example, John is at the same time a CMVC superuser, a releaseLead for rel-A and a normal developer for rel-B; he can add level members to a level in rel-A, but he should not do it for rel-B. However, if John is always using CMVC as a superuser, then it could be possible that John could successfully add, by accident, a level member in rel-B. If John only uses the CMVC superuser id for superuser-related tasks, and uses a normal userid for the majority of his CMVC tasks, then John would be prevented by CMVC to add a level-member for rel-B.

Based on our experience, we recommend that all the CMVC superuser ids should be easily identifiable, such as by adding the prefix "su_" to the normal CMVC user id. In this example, John would have his normal userid "john" and his CMVC superuser id "su_john". In that way, John can easily known when he is working as a superuser with CMVC.

HOW TO MODIFY AN EMAIL THROUGH THE BACKDOOR THAT HAS MORE THAN 15 CHARACTERS

Question:

I am trying to enter an email address that has more than 15 characters and I get the error message 0010-149.

Is there a fix for this problem? Is there a workaround?

Answer:

In CMVC 2.3.0.18 we fixed a limitation in the handling of the userid of the email address (the field before the @ "at" sign). The problem was that only the first 15 characters were accepted, and if this field had more than 15 characters, the message 0010-149 was shown.

- Solution:

  To upgrade to the CMVC version 2.3.0.18 or later.

- Workaround:

  If the migration to the latest CMVC release is not feasible at this time, then in some cases, it might be necessary to modify the address field for a user that has more than 15 characters (such as: 12345678901234567890@host) and it is possible to do that by bypassing CMVC and directly using the SQL statements. By the way, this is the method used by the 'mkdb' when creating the CMVC database.

  Note: Warning: before you follow the steps below, make sure that you have a backup of your database and of your CMVC family. If you make a mistake in the SQL statements, you may corrupt the database, and you will need to restore the latest backup.

  1. Logon as the CMVC family administrator.
  2. Connect to the database, such as:

     ```
     db2 connect to <family>
     ```

  3. Go into interactive SQL; in DB2 (command line processor) type:

     ```
     db2
     ```

  4. Find out the internal userid for the CMVC user whose address needs to be modified (example is shown below)

     ```
     db2 => select id,name,address from Users where login='doe'
                                                            ***
                                         put the user name here
     ```

     A sample output is shown below; you need to validate the name and address fields.

     ```
     ID      NAME                    ADDRESS
     ------- ----------------------- -----------------------------
     10      John Doe                short@carvm3
     ```

     Note that the current example complies with the limit of 15 characters in the first field of the address (before the @, "at" sign).

  5. Issue the following modify command (one single line), where the id field is the id found in step 4 (first field; in the example is 10):

```
update Users set address='12345678901234567890@carvm3' where id=10
                          ********************                  **
                          place here your value           user id
```

You can specify a value that has more than 15 characters in the first field (before the @ "at" sign); in the example there are 16 characters.

6. Commit the transaction (otherwise the value will not be seen by CMVC), by doing:

   ```
   commit work
   ```

7. Repeat step 4, and verify that the data is correct:

```
db2 => select id,name,address from Users where login='doe'
                                                      ***

ID      NAME                    ADDRESS
------- ----------------------- ----------------------------
10      John Doe                12345678901234567890@carvm3
```

Note that the new value has more than 15 characters in the first field of the address.

8. Terminate the DB session:

   ```
   terminate
   ```

ERROR 0010-655 WHEN TRYING TO DELETE A USERID

Question:

I am trying to delete some userids and I am getting the message:

```
0010-655 You cannot delete user xxx.  This user currently has
         access.  You must delete all access for this user
         before this user can be deleted.
```

Answer:

The problem is that the user is a member of one or more authority access groups, and you have to first delete the user from these groups:

1. Using the GUI, from the Authority Access window, use the Open List or Filter dialog to find out all the access groups that the desired userid is a member of.
2. Using the GUI, from the list of items returned by the previous step, highlight the items and invoke the delete action.
3. Then you may try to delete the user again.

BROADCASTING A MESSAGE TO ALL ACTIVE CMVC USERS

Question:

Is there an easy way to send a message to all active CMVC clients?

It could be necessary in case of an unplanned shutdown or to announce that a general problem has been solved.

Answer:

You may want to use the sample /usr/lpp/cmvc/samples/sendMailTo.

AUTHORITY ACCESS

COMPONENT OWNER CANNOT GIVE OBVIOUS AUTHORITY TO OTHER USERS

Question:

Component owner cannot give obvious authority to other users

Answer:

In CMVC 2.3.0.0 we updated the access authority groups to fill a gap. The componentlead is now a superset of developer+, and the releaselead is a superset of builder.

Originally, the shipped authority groups had some holes in the authorities which caused confusion, such as the fact that a developer+ has more FixYZ actions than the componentlead, which causes an error when a component owner tries to give authority of developer+ to a new user (the error says that the componentlead cannot give an authority that the component lead does not have, such as the FixYZ).

To correct this problem the following actions were added and are applied automatically in new families, but which need to be added manually into existing families.

1. Add to componentlead the following actions in authority.ld:

   ```
   DefectModify
   FixActive
   FixComplete
   FixDelete
   ```

2. Add to the releaselead the following actions in authority.ld:

   ```
   LevelAssign
   LevelCheck
   LevelCommit
   LevelComplete
   LevelCreate
   LevelDelete
   LevelModify
   MemberCreate
   MemberDelete
   SizeCreate
   ```

3. Stop the cmvc daemons and run chauth to reload the contents of the authority.ld file into the appropriate table in the database.

CONFUSION ABOUT COMPONENT OWNER AUTHORITY

Question:

I am confused about what a component owner can do. The CMVC User Reference states that the component owner can do many things implicitly, such as AccessCreate. That does not seem to be true. I have to give the component owner componentlead before she can add access for someone else. Am I reading something wrong or have we got something set up wrong?

Answer:

A non-superuser with AccessCreate authority may not grant any authority that includes any action that she/he does not explicitly have themself. In short, you ca not give a CMVC authority that you do not have.

The thing that makes this confusing is that as an owner you have certain implicit authorities but you cannot give these to someone else unless they are also granted to you explicitly.

CANNOT LOOK AT THE CONTENTS OF A FILE WITH GENERAL AUTHORITY

Question:

I am trying to view the contents of a file under general authority and I am getting the error:

```
0010-183 You are not authorized to perform action FileExtract
         on the specified file.
0010-184 You are not authorized to extract a copy of file
         x (x = file name) associated with release
         y (y = release name)
```

Answer:

In the default access authority groups shipped by IBM, the FileExtract action, which is the one that you need to look at the contents of a file, is not part of the general authority group.

However, the FileView action is part of the general authority group and this action will give you the meta-data that CMVC has about that file (such as version number, date last updated, etc).

You have some alternatives:

- Use an access authority group that has the FileExtract action, such as "developer".

This is the recommended solution.

- Add a new access authority group, such as "extractor", that will have this FileExtract action.

  Then, you can add this access group to those users who need to view the contents of files, even if their authority is general. CMVC handles the union of all the cumulative access groups for a user for a given component.

- You could add the FileExtract to the "general" access group.

  The drawback is that you may dilute the granularity of the access by components; that is, you may give too much authority to all the CMVC users.

HOW TO SPECIFY A READ-ONLY ACCESS TO FILES IN A COMPONENT STRUCTURE

Question:

Has anyone created an authority group which allows a user read-only access to files in a component structure?

Answer:

CMVC adds up all the authority actions from all the authority groups for a user. The result is list of all the authority actions that are valid for that user. For example, if a user belongs to two access lists (general and componentlead) in a given component, then the authority actions from general are combined with the authority actions from componentlead to provide the complete list of the authority actions for that user.

This means that a user could have very few authority actions in the root component (such as general) and then more authority actions in children components.

A corollary of the above is that you could define a new authority group such as "viewer" which would have ONLY few authority actions, such as "FileView" (to view the details) and "FileExtract" (to view the contents). Then you could add this authority group to any use and this will allow the user to ONLY view files. Of course, if the user has more FileXXX authority actions, then the user could create/modify files.

The nice thing about this approach is that you do NOT need to modify the existing authority groups.

Another possibility is add the desired action to the "general" authority group, and ensure that all users belong to this group.

WHAT IS THE ROLE OF THE CMVC USER ID "INHERITEDACCESS"?

Question:

The "InheritedAccess" userID is created automatically when a new CMVC family is created. What is the role for this userID? Could I delete it?

Answer:

Please, do not delete nor change the names of the following CRITICAL objects in CMVC, which are automatically created during the mkdb (the creation of the CMVC family database):

```
Component: root
User:      InheritedAccess
```

The "root" component is the top component for all active components.

The "InheritedAccess" user id is the visible aspect of the implementation (in CMVC Version 2) of the function to restrict the access in children components.

Note: Without these objects, the CMVC family will not work properly.

HOW DO I KNOW IF THE AUTHORITY HAS BEEN RESTRICTED IN A COMPONENT?

Question:

How do I know if the authority has been restricted in a component?

Answer:

Use Report -view accessView to view access for a component or user. The ***type*** column indicates the type of access (granted or restricted).

The userLogin ***InheritedAccess*** indicates that the authority group is restricted for all users inheriting this access.

Example output from Report -view accessView

```
compName userLogin       userName   userArea authorityName type
-------- --------------- ---------  -------- ------------- ----------
demotest jan             Jan Eaton  Testing  componentlead granted
demotest dave            Dave Iyers Develmt  developer+    granted
demotest pat             Pat Anders Mgmt     manager       granted
demotest pat             Pat Anders Mgmt     projectlead   restricted
demotest InteritedAccess                     projectlead   restricted
```

EXAMPLE OF INHERITED ACCESS IN A COMPONENT HIERARCHY

Question:

Example of inherited access in a component hierarchy

Answer:

An example of inherited access in a component hierarchy is shown below:

In the above diagram, pat is the project leader for objects managed by the code component and jan is the project leader for objects managed by the docs component. The component demotest is a child of both the code and the docs component.

Any access authority configured for these components will be inherited by the demotest component unless it is restricted.

If we just wanted to restrict pat from inheriting projectlead authority at the demotest component, then we could do so by specifically restricting pat to the projectlead authority.

If we wanted to restrict jan from inheriting projectlead authority, we could also do so in the same manner. Eventually this becomes a cumbersome process and we may want to establish that all users inheriting the projectlead authority are restricted at demotest.

CAN THE BASE AUTHORITIES BE RESTRICTED?

Question:

We want to restrict the base authority of specific users against a specific component. For example, we do not want certain users to open a defect against certain components (the DefectOpen is a base authority).

Answer:

```
                      +------+
                      | root |
                      +------+
                     /        \
                    /          \
               +------+      +------+
  pat :        | code |      | docs |  jan : projectlead
  projectlead  +------+      +------+        (granted)
 (granted)         \            /
                    \          /
                    +----------+
                    | demotest |   jan : componentlead (granted)
                    +----------+   dave : developer+ (granted)
                                   pat : manager (granted)
                                   pat : projectlead (restricted)
                                   InheritedAccess : projectlead
                                                     (restricted)
```

Figure 3. Example of inherited access

The base authorities cannot be restricted.

The Chapter 4 of the CMVC Users Reference say the following about the actions that have base authority:

```
"...
a base authority that can be performed by all users in the family"
```

ADDITIONAL DETAILS ON RESTRICTING AUTHORITY ACCESS

Question:

Additional details on restricting authority access

Answer:

The component owner, superuser or user with ***AccessRestrict*** authority can restrict access for users at a component.

Restricted authority is ***not*** inherited by child components.

Restricted authority does not affect implicit and superuser authority.

User whose access is restricted is notified. Subscribers to ***AccessRestrict*** are notified. When -inherited is used, only subscribers are notified.

When a family is created, a special user "InheritedAccess" is added to handle the restriction of authority access.

The id for this user is 0 in the Users table and must remain as 0 for the function to work correctly.

Each user on the access list of a component can have only one entry for each authority group. Therefore, if the user has the authority group 'developer' listed as a granted authority for a component, the server will prevent the addition of the authority group 'developer' as a restricted authority type for this user. This current authority group would have to be deleted and then the authority group can be restricted for the user.

If a user has multiple authority groups defined for a component where some are granted and others are restricted, then the user's access authority for that component will be the sum of all authority groups and the restricted actions will take precedence over the granted actions.

CLARIFICATION ON FILECHECKIN AND FILEUNDO AUTHORITY ACTIONS

Question:

Clarification on FileCheckIn and FileUndo authority actions

Answer:

- The FileCheckIn authority gives a person the power to check in a file EVEN if the file is locked by someone else.
- The FileUndo action for a specific change can be done by the user who did that change, even if this action is not explicitly given to that user.

CAN I SPECIFY FOR FILES: RO ACCESS TO ALL RW ACCESS ONLY TO CREATOR?

Question:

We have files in CMVC that all our group needs to read, but that should only be changeable by the creator. Can CMVC do this?

Answer:

CMVC does not have access restrictions at the File level, only at the Component level.

Thus, you may want to expand your component hierarchy and distribute your files accordingly. For example if all files for the GUI can only by modified by Sue, then move all those files into a new component and name Sue as the owner of that component (or at least, you give Sue the authority for FileCreate and FileModify, such as developer+).

Remember that the access lists are inherited from a parent component to a children component, and that the access lists are merged and the total access is the union of the individual access lists.

SPECIAL REPORTS

HINT WHEN USING % IN QUERIES IN REXX FILES

Question:

I am having problems with my REXX files in which the % wildcard for SQL statements is not properly replaced.

Answer:

The % character in REXX or in the OS/2 and DOS batch languages is used to denote an environment variable, such as %PATH%. However, if you want to use the % character for SQL queries, then you need to specify it double, as: %%

HOW TO LIST ALL FILES CHECKED IN BETWEEN 2 DATES

Question:

I would like to have a report that lists all the files checked in between two dates even if they are changed after this period.

Answer:

You could just use the versions table in a subquery:

```
Report -view FileView -where "sourceid in ( select sourceid \
       from Versions where changeDate between \
       '95/01/01' and '95/02/01')"
```

Also, you may want to play with variations of this query for performance reasons, for example:

```
Report -view FileView -where "releaseName = 'myrelease' and \
       exists  ( select sourceid \
       from Versions where changeDate between \
       '95/01/01' and '95/02/01' and \
       sourceid = FileView.sourceid)"
```

HOW TO GET A LIST OF ALL COMPONENTS THAT HAVE FILES IN A GIVEN RELEASE

Question:

How to get a list of all components that have files in a given release.

Answer:

One possible query is shown below:

```
Report -view compView -where "name in (select compName \
       from FileView where releaseName = 'alpha' and \
       dropDate is null)"
```

TO GET NUMERIC SORTING, USE VERSIONID INSTEAD OF VERSIONSID

Question:

The order of the items using versionSID seems to be incorrect in the following report:

```
Report -raw -view changeView -where "releaseName in ('relname') \
  and pathName in ('pathname') order by versionSID"
```

The order is:

```
  1.1
  1.10
  1.11
  1.2
  ...
  1.9
```

The expected order was 1.1 through 1.9, followed by 1.10 through 1.11.

Answer:

The versionSID (SID field in the versions table) is a variable character field so the Report is behaving correctly.

Instead, try ordering by versionid, because this field is numeric (it is generated from the Sequence table), it should provide you with the ordering that you were expecting.

WHERE DO THE NUMBERS FOR DEFECTS AND FEATURES COME FROM?

There are three "sequences" used in CMVC which are used to determine what internal "ID" numbers are assigned to any entry in a table that must be uniquely identified. The "numbers" assigned as names for defects and features are the most obvious. Each of these sequences is stored as a separate row in the family's SEQUENCE table. Once a number is used from one of these rows, the value is incremented.

Here are the names of each row and what they are used for:

defects The default name for any defect or feature. Since these numbers are very visible, this is the only use of this number.

One minor side effect of the fact that the ID for a defect/feature is also the default name, is that if you change the name to something like "Defect1413" instead of the default name "1413", then the number 1413 is still used up. As a result, the next defect will have a default name of 1414.

sourceId As explained previously (and in much more detail in the the TR 29.2297 "CMVC frequently asked questions: TRCMF-VC version control and database synchronization issues"), the sourceId points to where a file is stored in an SCCS archive in the VC tree.

This number is incremented each time a new file is created. The limit of six digits in a numeric field for some databases (4 for path plus 2 for SCCS archive filename) is the reason why CMVC only supports 1,000,000 files. In other words, when sourceID reaches 999999, then the last archive vc/9/9/9/9/s.99 is created.

general All other records in database tables that have a unique identifying key (id) use this number.

Because these keys must be unique, we strongly warn users that creating or deleting any record that contains a column with a suffix name of "id", (such as releaseId), risks corrupting the entire family. Records in tables that have columns named like releaseId, point to these unique records.

Further, these keys must be unique in order for all of the database views that are used by the GUI and Report command to display the correct values.

WHY DOES "FIELD <> 'VALUE'" NOT INCLUDE NULL?

Question:

Why does "field <> 'value'" not include null?

I queried the defects in my CMVC family for all the defects that do not have a certain value in the Release field; let's say that I got 100 records. Then I queried the defects that have a certain value and let's say that I got 50 records. However, I found out that there are really 175 defects in my CMVC family and the above actions count for only 150 of them. What happened to the missing 25 records?

Answer:

A null value is a value that forms a 3rd category when the equal/no-equal operator is used.

- Values that are equal.

  In the example above, there 50 records of this kind.

- Values that are not equal.

  In the example above, there are 100 records of this kind.

- Values that do not exist, that is, that are null.

  In the example above, there are 25 records of this kind.

Therefore, if you use the <> or the = operators, the resulting query will not include the values that are null.

SORTING DEFECT/FEATURE NAMES

Question:

If you do Report -view DefectView -where "order by name", the output would look like the following:

```
1
10
100
...
2
20
200
...
```

Instead of the expected result:

```
1
2
10
20
100
200
```

Answer:

This is because the defect name is now an alphanumeric field and it is left justified (it was a numeric field in CMVC V1). This was added to allow you to change the name of a defect.

To get the proper order of the defect names, use "order by addDate, name".

USE OF 1=1 AS ARGUMENT TO QUERY FOR ALL INSTANCES OF AN OBJECT

Question:

Use of 1=1 as argument to query for all instances of an object

Answer:

- You can use the argument "1=1" to query for all objects of the same kind.

  For example, to query for all users:

  - You can use the line command:

    ```
    Report -raw -view  Users  -where "1=1"
    ```

  - You can enter "1=1" (without the double quotes) in the entry field for the query in the Open List dialog (UNIX) or Filter dialog (OS/2).

- Another way is to use the following argument:

  ```
  Report -raw -view Users -where " login like '%' "
  ```

HOW TO SPECIFY DATES IN A QUERY

Question:

How to specify dates in a query.

Answer:

CMVC uses the format YY/MM/DD for the dates. Please notice that the year has only 2 digits. In TeamConnection we have fixed this potential problem by using 4 digits.

If you want to query all the defects that were opened since the beginning of April, 1996, you can do the following from the GUI:

1. Open the "Open List" or "Filter" dialog.
2. Locate the field that has dates, such as "Add Date".
3. Select the proper operand. In this example is greater-than ">".
4. Enter the date, such as April 1, 1996: 96/04/01
5. Apply or execute the query. A copy of the complete command that used this query is stored at the end of the CMVC.LOG file. See the GUI settings for the proper name and location for this file.
6. In this case, the query from the Open List for Defects will be:

   ```
   addDate > '96/04/01'
   ```

   Since the date also includes the time stamp in the string, you can also do the query by using something like this:

   ```
   addDate like '96/04/01%'
   ```

7. Then, the corresponding command at the bottom of the log will be:

   ```
   Report -raw -view  DefectView  -where "addDate > '96/04/01'"
   ```

HOW DO I USE A REPORT COMMAND IN A SHELL SCRIPT?

Question:

How do I use a Report command in a shell script?

Answer:

The default output format of the Report command is, in general, not appropriate for parsing in a shell script. The Report command provides a "-raw" option for most queries in which the fields are separated with a pipe or vertical bar (|) symbol. This makes for much easier parsing.

A shell script can use "awk" or "cut" to easily extract specific columns from a query. For example, to extract the second column from a report, you can do the following:

```
Report ... -raw | cut -d'|' -f2
```

NOTIFICATION

WHO GETS NOTIFIED WHEN A CMVC ACTION OCCURS?

Question:

Who gets notified when a CMVC action occurs?

Answer:

When a CMVC user performs an action, all of the users are notified according to the table in the Notification chapter in the CMVC User Reference.

There is one exception, the user who actually performs the action is NOT notified because that user did the action and thus, CMVC is not going to give the user an additional notification that the user did it.

Also, if there are several CMVC users with the same mail address, then only one of these users will get notified. Thus, if you have more that one CMVC user id with the same mail address and some of these CMVC user ids are involved in the notification, you will not get additional notifications for each of them.

HOW TO REMOVE SPECIFIC NOTIFICATION MESSAGES FROM CMVC

Question:

A user in our CMVC family does not want to receive CMVC notification messages whenever there are updates in the family for which the user is to be notified. Is there something we should do?

Answer:

There is no middle ground with the notifications from CMVC, that is, you cannot customize to receive certain messages but not other messages.

Some workarounds are shown below:

- You cannot remove the address from the user ID so that the user no longer receives these messages, because the address field is a required one. However, you could change the address to an email account that does not exist or that is created just for this purpose, such as

  ```
  nobody@hostName
  ```

- You could create a "dummy" userid in your system that can be used as the recipient of any CMVC messages that other users do not want to receive.

  Of course, the real CMVC users will not be able to get any messages from CMVC at all.

  In a sense, this "dummy" user id is the equivalent to having a null device such as /dev/null.

SUGGESTION OF USING ANOTHER ACCOUNT TO RECEIVE MESSAGES IN CMVC

Question:

Suggestion of using another account to receive messages in CMVC.

Answer:

It you are a componentLead or a releaseLead and you receive a lot of notifications from CMVC, you may want to consider creating another system account for yourself whose primary purpose would be to receive those notifications. In that way, these notifications will not clutter the mail of your primary account.

Once this secondary account is created, then you can simply modify the address of your CMVC user id to reflect the new secondary account.

APPENDIX A. BIBLIOGRAPHY

For more information on how to use CMVC, you can consult the following manuals and publications:

SC09-1596-01	IBM CMVC Client Installation and Configuration
SC09-1597-01	IBM CMVC User's Reference
SC09-1631-02	IBM CMVC Server Administration and Installation
SC09-1633-00	IBM CMVC Concepts
SC09-1635-01	IBM CMVC Commands Reference

The following Redbooks offer practical advice on CMVC:

GG24-4178-00	Did you say CMVC?
GG24-4345	CMVC: Customer's Perspective

The following technical reports describe in detail useful hints on using CMVC:

29.2169	How to use CMVC with National Language Support (NLS) and Double-Byte Character Sets (DBCS)
29.2180	How to do routine tasks in the OEM Platforms for CMVC
29.2183	Using CMVC for Products with Multiple National Language Versions (NLVs)
29.2232	How to do migration tasks with CMVC
29.2244	How to build and package the CMVC client for Windows 3.1
29.2245	How to build and package the CMVC client for OS/2
29.2253	Comparison between CMVC 2.3 and TeamConnection 2
29.2254	Migrating from CMVC 2.3 to TeamConnection 2
29.2268	CMVC frequently asked questions: license management with NetLS/iFOR
29.2269	How to build and package the CMVC server/Client for UNIX
29.2296	CMVC frequently asked questions: files, releases, levels, and tracks
29.2297	CMVC frequently asked questions: version control and database synchronization issues
29.2298	CMVC frequently asked questions: hints and tips on using databases and OEM platforms
29.2299	CMVC frequently asked questions: client for UNIX, OS/2 and Windows 3.1
29.2321	Comparison between TeamConnection 2 and CMVC 2.3 (2nd Edition)
29.2322	CMVC/TeamConnection: How to cope with dynamic IP addresses
29.2323	CMVC frequently asked questions: handling file, release and level extract problems

HOW TO GET ELECTRONIC COPIES OF MANUALS AND TRS

Many of the manuals and technical reports mentioned in this document can be downloaded as follows:

- From the IBM intranet (only for IBM employees).
- From the Internet (open to everyone).

IBM Intranet

Web Home Page

You can access the CMVC Service/Development Home Page at:

```
http://tc-cmvc.raleigh.ibm.com/cmvc
```

From the index at the top of the page, select:

- Technical Reports and related tools
- Documentation in ASCII text files

FTP

You can download the code from our internal FTP site, by doing:

1. ftp tc-cmvc.raleigh.ibm.com
2. login as 'anonymous' and for password give your email address.
3. cd e:\cmvc\doc\tr
4. binary
5. get fileName
6. quit

Internet

Web Home Page

Not available.

FTP

You can download the code from our external FTP site, by doing:

1. ftp ftp.software.ibm.com
2. login as 'anonymous' and for password give your email address.
3. cd ps/products/cmvc/doc
4. binary
5. get fileName
6. quit

APPENDIX B. COPYRIGHTS, TRADEMARKS AND SERVICE MARKS

The following terms used in this technical report, are trademarks or service marks of the indicated companies:

TRADEMARK, REGISTERED TRADEMARK OR SERVICE MARK	COMPANY
IBM AIX OS/2 CMVC DB2/6000, DB2 TeamConnection	IBM Corporation
PostScript	Adobe Systems Incorporated
HP-UX	Hewlett-Packard Company
Microsoft Windows MS-DOS	Microsoft Corporation
UNIX	X/Open Co., Ltd.

END OF DOCUMENT