CMVC frequently asked questions: migration of families Edition: 01-Dec-1999

Document Number TR 29.3114

Angel Rivera, Lee Perlov, Clifford Meyers

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

ABSTRACT

This technical report provides a collection of hints and tips for CMVC family administrators that want to migrate CMVC families. Some of the scenarios are:

- Changing the name of an existing CMVC family.
- Migrating a CMVC family from one host to another.
- Migrating a CMVC family from one DB2 instance to another.
- Migration to CMVC 2.3.1 which is Year 2000 ready.
- Preparing to migrate to VisualAge TeamConnection.
- Migrating data from library systems other than SCCS into CMVC.
- Miscellaneous migration issues.

ITIRC KEYWORDS

- CMVC
- VisualAge TeamConnection
- Migration

ABOUT THE AUTHORS

ANGEL RIVERA

Mr. Rivera is an Advisory Software Engineer and team lead for the CMVC development. 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 an Advisory 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.

CLIFFORD J. MEYERS

Mr. Meyers is an Advisory Software Engineer and the technical team leader of the TeamConnection packaging and test team. He joined IBM in 1985 and worked in the AIX/ESA development organization before joining the TeamConnection team.

CONTENTS

FIGURES

INTRODUCTION

This technical report provides a collection of hints and tips for CMVC family administrators that want to migrate CMVC families. This TR focuses on the latest modification of CMVC, version 2.3.1.4 which is Year 2000 ready. However, in the case of further refreshes to CMVC 2.3.1, it is recommended that you check periodically the CMVC external ftp site:

`ftp://ftp.software.ibm.com/ps/products/cmvc/README.supported.combinations.txt`

Also, the most current version of the non Year 2000 ready version of CMVC is 2.3.0.25.

This TR is organized as follows:

- Chapter "General information" on page 5 provides some overall recommendations when performing migration activities for the CMVC family.

  It also describes how to find out the version of the CMVC family server (cmvcd) and how to access the database management system (DBMS).

- Chapter "Migrating to CMVC 2.3.1 (which supports the Year 2000)" on page 19 describes how to migrate an existing CMVC 2.1, 2.2 or 2.3.0 to the new 2.3.1 which is Year 2000 ready. It also describes the Year 2000 readiness of CMVC 2.3.1: prerequisites and how it was tested.

- Chapter "Cloning/migration of a CMVC family from one host to another" on page 25 describes how to migrate (clone) a CMVC family from one host to another when the source host has the same operating system and DBMS as the target host.

- Chapter "Migration tips when moving from one version of CMVC to another" on page 31 provides procedures for performing specific migration tasks when moving a CMVC from one version of the operating system or DBMS or CMVC, to another version.

- Chapter "Preparing to migrate to VisualAge TeamConnection Version 3" on page 61 provides a brief description of VisualAge TeamConnection Enterprise Server Version 3, the successor of CMVC, and provides some recommendations for the migration of a CMVC family into TeamConnection.

- Chapter "Dealing with common migration problems" on page 63 provides guidelines on how to deal with common migration problems.

- Chapter "Renaming and moving a CMVC family" on page 71 describes a procedure that can be used to rename an existing CMVC family in the same host.

  This procedure can also be used for migration of CMVC families when it is desired to change operating systems, versions of CMVC, and/or versions of a database in one step. However, it takes longer than following the procedures documented in the CMVC Server manual.

This procedure is particularly useful when you wish to move data from the system (default) database in Oracle to separate database files.

- Chapter “Migrating data from library systems into CMVC” on page 77 describes how to migrate data from SCCS into CMVC. It provides a general overview of the tasks to be done when migrating from other library systems.
- Appendix Appendix A, “Obtaining the migration shell scripts and latest CMVC code” on page 83 describes how to get the tools mentioned in this TR and the latest CMVC code.
- Appendix Appendix B, “Migration shell scripts” on page 85 describes the tools mentioned in this technical report, and provides instructions on how to get them.

This technical report covers new information that has been gathered by the CMVC development and service support team since the publishing of the CMVC 2.3 manuals.

ACKNOWLEDGEMENTS

Many of the questions and answers that are compiled in this technical report were obtained from the TEAMC and CMVC forums 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!

We want to thank in particular the following co-workers:

- Kevin Postreich, TeamConnection team, IBM RTP, North Carolina.
- Keith Purcell, OEM Lab in IBM RTP, North Carolina.
- Jim Austin, AIX Lab in IBM RTP, North Carolina.
- Gary Greig, CMVC support team, IBM Austin, Texas.
- Gary Warner, CMVC support team, IBM Austin, Texas.
- Brian Johnston, IGS CMVC support team, IBM RTP, North Carolina.
- Tom Bing, Bell South, Atlanta, Georgia.
- Dan Fricker, American Express, Phoenix, Arizona.

We want to thank Dodde Stark for editing this technical report.

HOW TO GET THE MOST UP TO DATE VERSION OF THIS TECHNICAL REPORT.

The most up to date version of this technical report can be obtained from the IBM CMVC ftp site at URL:

`ftp://ftp.software.ibm.com/ps/products/cmvc/doc/tr`

For the list of available technical reports, see the file README.index.txt.

CMVC 2.3.1 PROVIDES SUPPORT FOR THE YEAR 2000

CMVC 2.3.1 is a new version-release-modification of CMVC (branched from CMVC 2.3.0.24) which uses 4 digits to represent the years instead of the 2 digits used in 2.3.0 and previous versions. Thus, CMVC 2.3.1 provides support for the Year 2000. For more details see “Migrating to CMVC 2.3.1 (which supports the Year 2000)” on page 19.

GENERAL INFORMATION

This chapter provides some overall recommendations when performing migration activities for the CMVC family.

OVERALL RECOMMENDATIONS

It is strongly recommended that you follow these suggestions:

- Make a complete backup of your CMVC family (both the file system of $HOME and the database) BEFORE starting the migration process. This will be your baseline backup. For more details see “How to make a complete backup of your CMVC family” on page 9.

  In case the migration process does not complete successfully, you can restore the family from the baseline backup.

- Try to break the migration process into manageable steps, in that way you can reduce the number of variables.

  For example, do not try to migrate in one single step a CMVC family from one CMVC version in one operating system version and database version, to another CMVC version in another operating system version and database version (too many variables at once!).

  That is, in order to make the target Host B as similar as possible to the source Host A, take smaller steps and ensure that CMVC is operational after each step.

- Make a complete backup of your CMVC family database AFTER each important step in the migration process.

  If the migration process involves several major steps, by making a backup of the CMVC family database after each major step, you could restore the CMVC family database from the previous step, if the current step did not complete successfully.

  If you do not make these intermediate backups, you may need to go back all the way to the first step in the sequence and restore from the original baseline backup.

  If you have obtained already a complete backup of the CMVC family database and of the $HOME file system, you only need to make a backup of the CMVC family database if you have not change the contents of files. In that way the backup process is expedited.

- Use separate user IDs and directories for the different "players". That is, install the CMVC code in one place (such as /usr/lpp/cmvc), the database code in another place, the DB2 instance in another place, and the CMVC family in another place.

  To avoid severe maintenance problems, it is strongly suggested to NOT add a CMVC family inside the directory of the CMVC code or the database code. The problem is that if

you store a CMVC family inside /usr/lpp/cmvc and then you decide to remove the CMVC code, accidentally you may be deleting also the complete CMVC family!

By the same token, it is recommended to not use the same user ID for both the DB2 instance and the CMVC family.

- It is highly recommended that you use the same name for the CMVC family user ID and the value for CMVC_FAMILY. This will help avoid the problem with DB2 when trying to restore a backup file. For more details see “How to restore a CMVC family from a complete backup” on page 14.

DOCUMENTATION FROM CMVC VERSION 2 RELEASE 3 ABOUT MIGRATION

The primary source for CMVC documentation on migration issues is in the "CMVC Server Administration and Installation: Version 2 Release 3" manual (IBM publication number SC09-1631-02). In this technical report we refer to this manual as the "CMVC Server manual".

The Version 2.3 of the CMVC Server manual contains many corrections to earlier releases; thus, use only the 2.3 version of this manual and do not use earlier releases.

This technical report does not cover the following migration issues which are covered by the CMVC Server manual:

1. Chapter 16, "Bringing SCCS Files under CMVC Control" describes how can migrate into the CMVC environment the text files that are already being developed with SCCS commands.
2. Appendix B, "Migrating to CMVC Version 2.3" covers the task on how to migrate from older versions of CMVC to the version 2.3.0.

   Notice that this appendix only covers up to version 2.3.0 and does not cover version 2.3.1.
3. Appendix C, "Converting Existing CMVC Server from ORACLE 6 to ORACLE 7" describes how to migrate from ORACLE 6 (which is not longer supported in CMVC 2.3) to ORACLE 7.
4. Appendix D, "Migrating to CMVC Server V.2.3.0 for DB2" covers the task on how to migrate from an existing CMVC family that is not using DB2 to a CMVC family that uses DB2.

   However, specific scenarios for migrating to DB2 Universal Database Version 5 (DB2 UDB V5) on AIX 4.2 are covered in this technical report.

Other related technical reports

- For details on how to configure DB2 UDB V5, see the TR 29.3076, "Configuration and Administration of DB2 Universal Database V5 by users of VisualAge TeamConnection V3". The main concepts discussed in the TR also apply to how CMVC uses DB2 UDB V5.

  It is available from:

  `ftp://ftp.software.ibm.com/ps/products/teamconnection/papers/trtc3db2.pdf`

- For details on how to use DB2 export and DB2 import to move a database from one place to another, see the TR 29.3088 "Moving a VisualAge TeamConnection Version 3 Family". The main concepts discussed in the TR also apply to a CMVC family.

  It is available from:

  `ftp://ftp.software.ibm.com/ps/products/teamconnection/papers/trmovedb.pdf`

- This technical report supersedes the TR 29.2232 "How to do migration tasks with CMVC".

HOW TO FIND OUT THE VERSION OF CMVC INSTALLED IN YOUR SYSTEM

Because of changes in functionality between CMVC 2.1, 2.2, 2.3.0 and 2.3.1, it is important to know exactly which version of CMVC you are using.

Starting with CMVC version 2.3, the appropriate version information is imbedded in the executable code and it allows you to determine what version of CMVC you are running.

The syntax for determining the version of the CMVC "cmvcd" executable file is shown below. The CMVC server, cmvcd, is used as an example because it contains also the information about the DBMS being used. Perform one of the following options:

- If you installed the product in the default directory:

  ```
  what /usr/lpp/cmvc/bin/cmvcd | grep cmvc
  ```

- If you are using the Korn shell:

  ```
  what $(whence cmvcd) | grep cmvc
  ```

- For AIX, SunOS and Solaris, use the following:

  ```
  what 'which cmvcd' | grep cmvc
  ```

  Note: The ' symbol is the "accent grave", located in the upper left corner in the English keyboard.

- For HP-UX use the following:

  ```
  what 'whence cmvcd' | grep cmvc
  ```

Note: The ' symbol is the "accent grave", located in the upper left corner in the English keyboard.

See the following sections for examples of the output.

Notes:

1. If the above procedure does not return a string, then you are running CMVC Version 1.x, or CMVC 2.1 or CMVC 2.2. Only by looking at the date stamps of the files might be possible to determine the version. For example, if the date is November 1993, then you have the last CMVC 2.2 version.
2. The CMVC server, CMVC client, and CMVC GUI are numbered independently. In this technical report we focus on the version of the executable for the CMVC server cmvcd.

Example of version information from the CMVC server

An example of the version information that can be obtained from the CMVC server is shown below:

```
cmvc     CMVC Version 2.3.1.4, (C) Copyright IBM Corp.,1993,1998
cmvc     5765-207, 5765-202, 5765-397, 5622-063
cmvc     98/09/10 SID=1.26.1.52
cmvc     Platform: AIX 4
cmvc     Database: DB2 UDB V5
```

Notice the version of the platform and of the database.

Example of version information from the executable CMVC GUI for Unix

An example of the version information that can be obtained from the executable CMVC GUI for Unix is shown below:

```
cmvc     CMVC Version 2.3.1.4, (C) Copyright IBM Corp.,1993,1998
cmvc     5765-207, 5765-202, 5765-397, 5622-063
cmvc     98/10/22 SID=1.1.5.4
cmvc     X11 R5
cmvc     Motif 1.2
cmvc     Platform: AIX
```

Notice the version of the Motif and X11 toolkits that were used to build the executable file.

Version information from Help -> On Version in CMVC GUI for Unix

An example of the version information that can be obtained from doing Help -> On Version in the CMVC GUI is shown below:

```
IBM Configuration Management Version Control
Client Feature for Motif
Version 2 Release 3 Modification 1 Level 4
```

WHICH TOOL TO USE TO ACCESS THE DATABASE

In this technical report we use concrete examples that use the database access tool (the SQL prompt) for a specific database. The tool to connect to a specific DBMS is:

DB2	db2 connect to $CMVC_FAMILY
Oracle	sqlplus $CMVC_FAMILY/$ORACLE_PASS
Sybase	isql -P $SYBASE_PASS
Informix	There are 2 options: • isql $CMVC_FAMILY • dbaccess $CMVC_FAMILY

Notes:

1. Where $CMVC_FAMILY represents the short name of the CMVC family, which is also the name of the actual database associated with the family.
2. Where "$ORACLE_PASS" and "$SYBASE_PASS" represent the password for the CMVC family and not the password for the DBMS user ID.

HOW TO MAKE A COMPLETE BACKUP OF YOUR CMVC FAMILY

Overall considerations for backup

The procedure described in this section is the recommended sequence for making a complete backup of your CMVC family. This backup could be the normal daily backup or the backup that is done before migrating the CMVC family. The following guidelines are used in this TR:

- Keep the process simple.
- Perform a full database backup; that is, no incremental backups.
- Perform an off-line backup; that is, the CMVC family should not be active. The advantage is that you do not have to deal with the transaction logs.

A very important point to keep in mind is that a CMVC family consists of the following two parts that need to be synchronized at all times and therefore they need to be backed up at the same time:

- The database where the CMVC objects are kept.
- The $HOME of the CMVC family user ID that has the vc tree (where the file changes are kept), the user exits, the configurable fields, and the configuration items (the *.ld files).

A common mistake is to only take the backup of the database or of the $HOME directory but not both. The problem arises when trying to restore from the backup files because the database will be out of sync with respect to vc tree.

The counterpart for the backup operation is the restore, which is described in “How to restore a CMVC family from a complete backup” on page 14.

Procedure for performing the backup

The recommended backup procedure is shown below:

1. Login to the family and stop the family servers. This backup procedure is an off-line backup and therefore, the database to be backed up must not be in use.
2. When doing the backup for the first time, create a directory such as $HOME/backup, where the backup file of the database will be kept.

   ```
   mkdir backup
   chmod 777 backup
   ```

3. Using the database utilities, make a backup of the CMVC family database and specify that the backup file should be kept in $HOME/backup.

   A summary of the backup procedures for the different DBMSs is shown in:

   DB2 See “Backup of DB2 databases” on page 11.
   Oracle See “Backup of Oracle databases” on page 13.
   Informix See “Backup of Informix databases” on page 11.
   Sybase See “Backup of Sybase databases” on page 12.

   Your database manual will provide the details on how to do this.

4. Place the backup file of the database in the HOME directory of the CMVC family, such as in $HOME/backup.
5. Use the utility "tar" (the utility "backup" is also available on AIX) to make a tar of the entire CMVC family account, which now includes the backup of the CMVC database.

   For example, to tar the family user ID file system into a file named /tmp/family.tar do:

   ```
   $ cd
   $ tar -cvf /tmp/family.tar *
   ```

For large families, you may want to combine this step with the next by specifying a tape device instead of /tmp/family.tar.

6. If you created a tar file, copy it into tape or another archival media.

Note: CMVC provides a sample backup utility for DB2 in "/usr/lpp/cmvc/samples/backupCMVC".

Backup of DB2 databases

For DB2 the sequence is:

1. Invoke the DB2 command line processor:

   ```
   $ db2 connect to $CMVC_FAMILY
   ```

2. Enter:

   ```
   $ db2 backup database $CMVC_FAMILY to $HOME/backup
   ```

3. Wait for the backup to complete. You will see the message:

   ```
   Backup successful. The timestamp for this backup image is: 19981124023858
   ```

   Now you have a backup copy of your database in the directory $HOME/backup.

4. Terminate the DB2 session:

   ```
   $ db2 terminate
   ```

Backup of Informix databases

1. Login as the CMVC family administrator ($LOGNAME):
2. Stop the CMVC family:

   ```
   stopCMVC $CMVC_FAMILY
   ```

3. Create a directory where to store the backup files:

   ```
   $ mkdir $HOME/backup
   $ chmod 777 $HOME/backup
   ```

4. Use the **dbexport** command to export the database for the CMVC family (in a directory $HOME/backup/$CMVC_FAMILY.exp):

   ```
   $INFORMIXDIR/bin/dbexport -o $HOME/backup $CMVC_FAMILY
   ```

5. Wait for the backup to be completed:

   ```
   dbexport completed
   ```

Backup of Sybase databases

This section describes how to create a dump device by using the **sp_addumpdevice**. and how to use the **dump database** command to dump the database to the device recently created.

For more details see the Sybase documentation, section "Dumping and Loading: SQL Server Scheme".

1. Login as the CMVC family administrator ($LOGNAME):
2. Stop the CMVC family:

   ```
   stopCMVC $LOGNAME
   ```

3. Create a directory where to store the backup files:

   ```
   $ $HOME/backup
   $ chmod 777 $HOME/backup
   ```

4. If using Sybase 11 you must start the Backup Server:

   a. login as the Sybase user id
   b. cd install
   c. RUN_SYB_BACKUP &

5. As the CMVC family user id, invoke isql:

   ```
   $ isql -Usa -P$SYBASE_SA_PASS
   ```

6. Run the following command to back up the family database. You need to replace **family** with the name of your family and specify the actual directory where you want the backup file to be saved.

   - For Sybase 4.9 or 10

     ```
     sp_addumpdevice "disk", dobackup,
     "/export/home/family/backup/family.bak", 2
     go
     dump database family to dobackup
     go
     ```

     Where the number 2 in sp_addumpdevice is the control type.

   - For Sybase 11

     ```
     sp_addumpdevice "disk", dobackup,
     "/export/home/family/backup/family.bak"
     go
     dump database family to dobackup
     go
     ```

     Wait until you get the following message that indicates that the backup procedure is done:

```
Backup Server: 3.42.1.1: DUMP is complete (database familyName).
```

Now you have a backup copy of your database in the file /export/home/family/backup/family.bak.

Notes:

1. You can also dump the database to a tape device by changing "disk" to "tape", "/home/family/backup/family.bak" to the device name of the tape and the controller number parameter 2 to another number which must be between 3 and 8 (tape volume) in the above command.
2. Use **sp_dropdevice deviceName** to drop dump devices.
3. To stop the Backup Server, as the Sybase user id, do the following:
   a. cd install
   b. ./showserver
   c. Identify the process id of the RUN_SYB_BACKUP process, such as 1234.
   d. Use "kill 1234" to terminate the Backup Server.

Backup of Oracle databases

1. Login as the CMVC family administrator ($LOGNAME):
2. Stop the CMVC family:

   ```
   stopCMVC $LOGNAME
   ```

3. Create a directory where to store the backup files:

   ```
   $ $HOME/backup
   $ chmod 777 $HOME/backup
   ```

4. Use the **exp** command to export the database to a file prior to backing up the HOME directory for the CMVC family. For example (in one single line)

   ```
   $ORACLE_HOME/bin/exp $ORACLE_DBA buffer=40000 \
   file=$HOME/backup/oracle.dmp grants=n indexes=y rows=y constraints=n \
   compress=y full=n record=n owner=$LOGNAME
   ```

5. Wait for the backup to be completed:

   ```
   Export terminated successfully without warnings
   ```

HOW TO RESTORE A CMVC FAMILY FROM A COMPLETE BACKUP

Once you have made a complete backup of the CMVC family (which includes both the $HOME of the family and the database used by the family) as explained in “How to make a complete backup of your CMVC family” on page 9, you need to follow this procedure to restore the CMVC family:

1. Login to the family and stop the family servers (if any). This restore procedure is an off-line procedure and therefore, the database to be restored must not be in use.

2. Use the utility "tar" (or "restore" if you used the "backup" utility on AIX) to unpackage or untar the tar file that contains the entire CMVC family account and the backup of the CMVC database.

   If the $HOME directory had other files or subdirectories, then you may see some messages indicating that the file permissions of the existing files did not allow for them to be replaced by the files from the tar file.

   To avoid these messages, it is suggested to perform the following command in a clean $HOME file system:

   ```
   $ cd
   $ tar -xvf /tmp/family.tar
   ```

3. If using DB2, ensure that the user ID and the database name in the target host is the same as in the source host.

   If they are not the same, then you will get a DB2 SQL0969 error when doing a restore due to the different authorization. This error message is extremely confusing!

   To find out the authorization in the target host do:

   a. db2 connect to $CMVC_FAMILY

   b. Look at the last 2 lines such as:

      ```
      SQL authorization ID   = BUILD
      Local database alias   = BUILD
      ```

      The first statement indicates the name of the user ID.

      The second statement indicates the name of the database.

      Note: It is highly recommended that you use the same name for the CMVC family user ID and the value for CMVC_FAMILY.

4. Using the database utilities, restore the CMVC family database and specify that the backup file is located in $HOME/backup.

   A summary of the restore procedures for the different DBMSs is shown in:

   DB2 See “Restore of DB2 databases” on page 15.
   Oracle See “Restore of Oracle databases” on page 17.

Informix See “Restore of Informix databases” on page 15.
Sybase See “Restore of Sybase databases” on page 16.

Your database manual will provide the details on how to do this.

Restore of DB2 databases

For DB2 the sequence is:

1. Login as the CMVC family administrator ($CMVC_FAMILY):
2. Stop the CMVC family:
   ```
   stopCMVC $CMVC_FAMILY
   ```
3. If the CMVC family does not exist yet (such as when migrating), then you need to run "mkfamily" and "mkdb -d" in order to create one. During the restore operation, the database from the backup file will overwrite the existing database, and this is fine.
4. Invoke the DB2 command line processor:
   ```
   $ db2 connect to $CMVC_FAMILY
   ```
5. Enter:
   ```
   $ db2 restore database $CMVC_FAMILY to $HOME/backup
   ```
6. Wait for the restore to complete. You will see the message:
   ```
   DB20000I The RESTORE DATABASE command completed successfully
   ```
7. Terminate the DB2 session:
   ```
   $ db2 terminate
   ```

Restore of Informix databases

Use the **dbimport** command to import the tables, indexes and views for the CMVC family.

1. Login as the CMVC family administrator ($LOGNAME):
2. Stop the CMVC family:
   ```
   stopCMVC $LOGNAME
   ```
3. If you are restoring a database into an existing one, then it is necessary to drop/delete the database:
   ```
   $ rmdb
   ```
   If the database exists, then the following error will occur when trying to restore the database from the backup file:

```
*** create database
330 - Cannot create or rename database.

100 - ISAM error:  duplicate value for a record with unique key.
```

4. If INFORMIX_DBSP is not set, run the following:

   ```
   $INFORMIXDIR/bin/dbimport -i $HOME/backup -l -ansi $LOGNAME
   ```

5. If INFORMIX_DBSP is set, run the following (in one single line):

   ```
   $INFORMIXDIR/bin/dbimport -i $HOME/backup -l -ansi -d $INFORMIX_DBSP \
   $LOGNAME
   ```

6. Wait for the restore to be completed:

   ```
   dbimport completed
   ```

7. The above commands will create a database with the name of the CMVC family with log (-l) and mode ansi (-ansi). The data is loaded into INFORMIX_DBSP or into rootdbs if INFORMIX_DBSP was not specified.

Restore of Sybase databases

Use the **load database** command to restore the database for the CMVC family.

For more details see the Sybase documentation, section "Dumping and Loading: SQL Server Scheme".

1. Always make sure that the destination database must be large enough to hold the amount of storage space that was actually allocated to the dumped database.
2. Login as the CMVC family administrator ($LOGNAME):
3. Stop the CMVC family:

   ```
   stopCMVC $LOGNAME
   ```

4. If appropriate, simulate the loss of the database and then recreate the basic database that includes the default configurable fields shipped by IBM:

   ```
   $ rmdb
   $ mkdb -d
   ```

5. If using Sybase 11 you must start the Backup Server:

   a. login as the Sybase user id
   b. cd install
   c. RUN_SYB_BACKUP &

6. Ensure that you have the appropriate dump device defined in Sybase. This dump device should have the backup of the database of the CMVC family.
7. Run the following command to restore the Sybase database:

```
$ isql -Usa -P$SYBASE_SA_PASS
1> load database family from dobackup
2> go
```

Where family is the name of the CMVC family.

8. In Sybase 11, wait for the following message that indicates that the restore was completed:

   ```
   Backup Server: 3.42.1.1: LOAD is complete (database familyName).
   ```

9. In Sybase 11, you need to have stop and restart the SYBASE servers in order to change the status from offline to online.

   Then you can restart the CMVC family.

Restore of Oracle databases

1. Login as the CMVC family administrator ($LOGNAME):
2. Stop the CMVC family:

   ```
   stopCMVC $LOGNAME
   ```

3. If you are restoring into the same CMVC family id, then drop the existing database:

   ```
   $ rmdb
   ```

4. Create an Oracle userid with the same name as your CMVC family. This userid has the password kept in the ORACLE_PASS environment variable. For example:

   ```
   $ORACLE_HOME/bin/sqlplus $ORACLE_DBA
   GRANT CONNECT, RESOURCE TO familyName IDENTIFIED BY oracle_pass;
   ```

   Where you need to provide the actual values for familyName (from $LOGNAME) and oracle_pass (from $ORACLE_PASS).

5. If you have tables stored in a different tablespace, alter the Oracle userid and make its default tablespace to be the one kept in ORACLE_TBLSP environment variable. For example:

   ```
   ALTER USER familyName DEFAULT TABLESPACE oracle_tblsp;
   EXIT
   ```

   Where you need to provide the actual values for familyName (from $LOGNAME) and oracle_tblsp (from $ORACLE_TBLSP).

6. Use the **imp** command to import the tables, indexes and views for your CMVC family. For example (in one single line):

   ```
   $ORACLE_HOME/bin/imp $ORACLE_DBA buffer=40000 \
   file=$HOME/backup/oracle.dmp commit=y show=n ignore=n grants=n \
   indexes=y rows=y destroy=n full=n fromuser=$LOGNAME touser=$LOGNAME
   ```

7. If you have indexes stored in a different tablespace, do not import the indexes, create them after the tables have been imported. For example (in one single line):

```
sed "s/TABLESPACENAME/$ORACLE_NDXSP/g" $CMVC_HOME/install/index.db | \
$ORACLE_HOME/bin/sqlplus $LOGNAME/$ORACLE_PASS
```

MIGRATING TO CMVC 2.3.1 (WHICH SUPPORTS THE YEAR 2000)

PREREQUISITES FOR YEAR 2000 READINESS

The CMVC 2.3.1 version-release-modification provides support for the Year 2000 by using 4 digits instead of 2 to represent the year. The new CMVC 2.3.1.1 was branched out from CMVC 2.3.0.24.

For the most up to date information about CMVC and the Year 2000 readiness, get the following files from the CMVC external ftp site:

```
ftp://ftp.software.ibm.com/ps/products/cmvc/README.year2000.txt
ftp://ftp.software.ibm.com/ps/products/cmvc/README.year2000.details.txt
```

Prerequisites for running CMVC 2.3.1

CMVC uses the Unix utility SCCS to provide the version control for the file changes. The original version of SCCS is not ready for the Year 2000: a file created in 19xx cannot be checked out in 20xx, nor a new file can be created in 20xx.

However, the Unix utility SCCS was been fixed to be ready for the Year 2000 in the versions of the operating systems shown below. We have successfully tested CMVC 2.3.1 in these environments by creating files in 1998 and then changing the system date to 2005 and then doing a checkout/checkin and creating a new file. This means that CMVC 2.3.1 and not CMVC 2.3.0 can ONLY work in the Year 2000 with these environments.

- AIX 4.2.1
- HP-UX 10.20, with patch PHCO_15215.
- Solaris 2.5.1, with patches 103612-38 and 103801-06.
- AIX 3.2.5: PTF U447712, APAR IX55509 and IX75948.

  CMVC 2.3.1 for AIX 3.2.5 is not officially supported because AIX 3.2.5 was officially discontinued in 31-Dec-1997. Thus, you need to migrate from AIX 3.2.5 to AIX 4. However, for completeness we are documenting the needed patches for SCCS to be Year 2000 ready:

As far as we know, the SCCS utility has not been fixed in the following operating systems and thus, CMVC 2.3.1 will not work correctly in the Year 2000:

- AIX 4.1.x, which is scheduled to be officially discontinued in 31-Dec-1998.
- Solaris 2.3, 2.4, 2.5.0 without the patches.

Finally, we have NOT tested CMVC 2.3.1 in the following platforms and thus, we cannot guarantee that the SCCS utility will properly work with the Year 2000:

- SunOS 4.1.3
- HP-UX 9
- HP-UX 10.01, 10.10

Note: All versions of the operating systems supported by CMVC need more fixes and patches in order to be completely Year 2000 Ready. The patches shown above are the ones that only fix SCCS. For more details see “Summary of main URLs with Year 2000 ready information.”

Summary of main URLs with Year 2000 ready information

All versions of the operating systems supported by CMVC need fixes and patches in order to be Year 2000 Ready. Of critical importance are the patches for the UNIX utility "SCCS", because without these fixes, you will NOT be able to checkout files in the Year 2000!

The following list of URLs indicate where you can get more information. This list is in HTML format, in that way you can copy and paste into an HTML file and then use a web browser to visit those sites and get the fixes.

```
<ul>

<li> <a href="http://www.software.ibm.com/year2000/papers/aixy2k.html">
AIX Year 2000 Workbook </a>, information about the patches that are
needed for all versions of AIX.

<li> <a href="http://wwwyr2k.raleigh.ibm.com/">
IBM Year 2000 Home Page - Product readiness database</a>

<li> <a href="http://www.software.hp.com/products/Y2K/corelist.html">
HP-UX Year 2000 Core Operating System Master Patch List</a>

<li> <a href="http://www.sun.com/y2000/faq.html">
Sun's Year 2000 FAQ </a>, this is a good place to start,
that has specific details.

<li> <a href="http://www.sun.com/y2000/cpl.html">
Sun's Year 2000 - Compliant Product List </a>, with links to get
the patches.

</ul>
```

MIGRATION STEPS TO UPGRADE TO CMVC 2.3.1

Note: If this procedure is not followed in a family that was migrated from CMVC 2.3.0, then there will be a mix of years, the old ones will have only 2 digits and the new ones will have 4, which will cause sorting problems.

To migrate a family that uses CMVC 2.3.0 or a previous version, follow these steps:

1. Stop the CMVC family and make a backup as recommended in “How to make a complete backup of your CMVC family” on page 9.
2. Migrate CMVC to 2.3.1:
   - If migrating from CMVC 1.x, 2.1, or 2.2, then follow the procedure explained in the Appendix B of the CMVC Server 2.3 manual.

     Then continue with step 3.
   - If migrating from CMVC 2.3.0, then it is just necessary to install CMVC 2.3.1.

     Then continue with step 3.
3. Run dbSetDate.

   For each CMVC family to be migrated, login to the family account and run the 2.3.1 migration shell script in order to modify all the instances for the date in the database from "yy/mm/dd" to "yyyy/mm/dd":

   ```
   /usr/lpp/cmvc/install/dbSetDate  <database> <family> <logfile>

   Where <database> is ORACLE7 (for ORACLE 7)
                       INFORMIX (for Informix)
                       DB2 (for DB2)
                       SYBASE (for Sybase)
   Where <family> is the name of your family
   Where <logfile> is the name of a file to append the SQL output
   ```

   For example, to update the dates for a family that uses DB2 and to store the SQL output in the file "sql.out" do the following:

   ```
   $ dbSetDate DB2 $CMVC_FAMILY sql.out
   ```
4. If using DB2, then you must issue the "db2bind" command after dbSetDate, for example:

   ```
   $ db2bind $CMVC_FAMILY
   ```
5. Perform a quick sanity check for the migrated family; some tests that you can do are to verify the new format for the year (yyyy) are:
   - List all users and verify the columns with dates.
   - View one user and verify the dates.
   - From a Open List (Filter) Users window, issue a query that lists all users created after a certain date, such as:

```
addDate > '1997/01/01'
```

- List other CMVC objects such as defects, files, releases, components and verify the dates.
- Do a release extract using the -date flag.
- If possible, do a release link using the -date flag.

  You could create a dummy release just to do the link and then undo the links and delete the release.

6. In case that you have not done it before, add the new indexes to improve performance that were added in 2.3.0.18 and which are listed in the file:

   `/usr/lpp/cmvc/doc/README.pubs`

   Note: These indexes are created automatically only when a new family is created. That is, if you created a family in 2.2 and you migrated it to 2.3.0.25 or 2.3.1.4, then the new indexes are not automatically added during the migration.

7. Make a backup of your CMVC 2.3.1 family. For details see “How to make a complete backup of your CMVC family” on page 9.

DETAILS ON HOW CMVC 2.3.1 WAS TESTED FOR THE YEAR 2000

The purpose of this section is to provide details on the implementation and testing of CMVC 2.3.1.4, which is the version of CMVC that is Year 2000 ready. For general details on the prerequisites, migration steps, and other high level information about CMVC 2.3.1, see “Pre-requisites for Year 2000 readiness” on page 19 and “Migration steps to upgrade to CMVC 2.3.1” on page 21.

- Change to the now() routine.

  The basic and most fundamental change between CMVC 2.3.0 and 2.3.1 is that the year is now represented using 4 digits (such as 1998, 2000) instead of only 2 digits (such as 98, 00). The update was done in the now() function which returns a date time stamp with the format "YYYY/MM/DD HH:MM:SS". Thus, any attribute in CMVC that deals with a date, uses the now() function: addDate, lastUpdate, dropDate, etc.

- No impact at all to the CMVC client

  The now() function is only used by the CMVC Server and not by the CMVC client. In fact, the CMVC client does not split or manipulates the date information, which it is treated only as a single string of characters. Fortunately, the string size for the date was big enough to accommodate the expansion from 2 to 4 digits. Thus, there is NO impact to the CMVC client (both line commands and GUI). This means that a CMVC 2.3.0 client can talk to a CMVC 2.3.1 server, and vice versa, a CMVC 2.3.1 client can talk to a CMVC 2.3.0 server.

  The only changes to the CMVC 2.3.1 client are purely cosmetic:

  - Update of the samples in the Help whenever a date is used.
  - Update of the version number in the Copyright information.

- Other changes in the server.

  The following functions in CMVC were affected by the readiness with the Year 2000:

  - Release -extract by date: now it parses the 4 digit representation of the year.
  - The dates in the audit/log file are now using the 4 digits.
  - Track and Level -commit: because they use the year for query, they needed to be updated.

- Miscellaneous (outside the scope of CMVC)

  The SCCS keywords that deal with dates (such as D, surrounded by %), represent the year with only 2 digits. CMVC has not control over it. The important point is that SCCS must have the necessary fixes/patches to allow it to handle the year 2000 correctly.

CLONING/MIGRATION OF A CMVC FAMILY FROM ONE HOST TO ANOTHER

This chapter describes how to migrate (clone) a CMVC family from one host to another.

Even though the main focus is when the target host has the same operating system and DBMS as the source host, this procedure can be used when there are differences in the version of CMVC, the operating system or DBMS between the hosts. (see “Migration tips when moving from one version of CMVC to another” on page 31 for some specific scenarios).

GENERAL PROCEDURE

Scenario:

1. Current CMVC server is on Host A (which is called "the source host" in this document).
2. We installed in Host B (which is called "the target host") the same version of the operating system, of the database management system and of CMVC used in source Host A.

   Note: The general recommendations in the in this chapter also apply when migrating from a source host to a target host and there are differences in operating systems and/or databases.
3. We want to copy the CMVC family from the source Host A on the target Host B.

Recommendation:

When moving a family from one host to another there are some common problems that can be avoided by following the suggestions below:

- In case you are migrating from the source host because you need to upgrade the operating system in that source host, then wait until the family is moved into the target host before upgrading your source host.
- Make sure that all of the CMVC characteristics are the same between the source Host A and the target Host B:
  - Same user ID in /etc/passwd.
  - Same group ID in /etc/group.
  - Same port number in /etc/services.
  - Same host alias in /etc/hosts.

  If you want, you may change these things later, after the family has been successfully moved.

- Keep in mind the general guidelines mentioned in “Overall recommendations” on page 5.

Procedure.

Perform the procedure in the following order:

1. Pre-migration steps in source Host A. See “Pre-migration steps to be done in source Host A.”
2. Migration steps in target Host B. See “Migration steps to be done in target Host B.”
3. Post-migration steps in target Host B. See “Post-migration steps to be done in target Host B” on page 28.

PRE-MIGRATION STEPS TO BE DONE IN SOURCE HOST A

1. Make any necessary host list changes for the new host, such as adding a hostlist for the superuser. If you do not do this now, then you will have to create a host list entry directly into the database after the migration. See :hdref=hostcre. for details.
2. Check the value of LANG and run the command **locale**.

   The new family account in Host B will need to match these values.

   When migrating from AIX 3.2.5 to AIX 4, see “AIX 4: necessary links if using the en_US locale” on page 32.
3. Stop the CMVC family. We suggest to use the "/usr/lpp/cmvc/samples/stopCMVC" sample to terminate the daemons and to clean up shared memory:

   ```
   $ stopCMVC $CMVC_FAMILY
   ```
4. Make a backup for the CMVC family; this includes both the CMVC user id directories and the database used by the CMVC family. For more details see “How to make a complete backup of your CMVC family” on page 9.

MIGRATION STEPS TO BE DONE IN TARGET HOST B

1. Database related tasks:
   - If using DB2, create a new DB2 instance owner and start the DB2 instance.
   - If using Oracle, Sybase or Informix, you may need to create separate files for table spaces to restore the database.
2. Login as root and create in Host B, a user ID with exactly the same values as the user ID of the existing CMVC family in Host A, for example same user ID, same group ID, same directory, etc.

If using DB2, then ensure that the CMVC account should use group id of the DB2 instance to be used.

Note: In case that you do not use the same userid and/or group, you will need to change the ownership of the files that are transfered from the old CMVC family user id to reflect the new user id and/or group id for the new CMVC family.

3. Modify the .profile for the CMVC user ID. Take into account the following:
   - Make sure that LANG environment variable is the same as in Host A.

     When migrating from AIX 3.2.5 to AIX 4, see “AIX 4: necessary links if using the en_US locale” on page 32
   - If using DB2, see “Warning when migrating a DB2 family from AIX 3 to AIX 4” on page 57.
4. Login as the CMVC user ID.
5. If using DB2, run the "locale" command to make sure that the correct locale is used.

   If any "LC_*" variables have a value of 'C', the locale may not be installed and the DB2 restore may not work. Use the "locale -a" command to find out which locales are available in your system.
6. Create a new CMVC family with the same setup as the old family:
   - Update the /etc/host and /etc/services to reflect the new family.
   - Create the subdirectories and files for a CMVC family:

     ```
     $ mkfamily
     ```
   - Create the database for the CMVC family (assuming that it uses the default configurable fields shipped by IBM):

     ```
     $ mkdb -d
     ```

     This will create the default configurable fields shipped by IBM. However, the configurable fields actually used by the source family will be restored when you unpack/untar the family tar file.
7. Copy from the source Host A the family tar file into the target Host B. In this example, it is assumed that this family tar file is copied into /tmp.
8. Unpack (untar) the family tar file and then restore the database. For more details see “How to restore a CMVC family from a complete backup” on page 14.

POST-MIGRATION STEPS TO BE DONE IN TARGET HOST B

1. Reorganize the indexes for the database. This step is optional but highly recommended. This is a good point in the sequence to reorganize and update the indexes for the database, to improve performance.

   If using DB2, use the command:

   ```
   db2reorg  $LOGNAME  /tmp
   ```

   The last step in db2reorg is to automatically invoke the command to bind the executable code to the DB2 instance:

   ```
   db2bind $CMVC_FAMILY
   ```

2. If the migration was from a CMVC 2.3.0 family or earlier (2.1.x or 2.2.x), then you need to execute the dbSetDate utility. For more details, see “Migration steps to upgrade to CMVC 2.3.1” on page 21.

3. Test the migrated CMVC family. Issue several CMVC commands to verify that the contents of the tables, such as users, components, files, is correct.

   Try to issue CMVC commands from the CMVC family user ID (normally a superuser ID) and from a non-superuser ID.

   Some specific queries that you may want to execute in both the source Host A and in the target Host B are shown below. If you are not using DB2, then use the appropriate SQL command for your DBMS.

   The following queries must execute successfully, without any warning or error messages:

   - Report -testServer
   - Report -view ReleaseView
   - db2 "select * from Sequence"
   - db2 "select id, compId, userId, relSize from Releases where id=0"
   - db2 "select id, previousId, sourceId, userId from Versions where id=0"
   - db2 "select id, releaseId, userId from Levels where id=0"

   In case of problems, see “Dealing with common migration problems” on page 63.

4. Extract a release from the source CMVC family and compare the result with an extraction of the corresponding release from the target CMVC family.

   Verify that the number of files is the same, verify that the contents of some specific files are the same, etc.

5. Make a backup of the new CMVC family. For more details see “How to make a complete backup of your CMVC family” on page 9.

6. Any user who perform level or release extracts will have to NFS export the target directory to the new Host B.

7. Either update the new family name in the domain name server or ask all users to update their /etc/hosts files.
8. If a user is using the VM client, you may need to install/start the vmkld daemon on the new server and create additional host list entries for each user.

Note: Do NOT use the CMVC cmvcarchive and cmvcrestore utilities for backups. These tools are designed to archive old releases and/or levels that will not be needed in the current family. If this archived information is needed, it can be restored into a new, hopefully temporary, family.

MIGRATION TIPS WHEN MOVING FROM ONE VERSION OF CMVC TO ANOTHER

This chapter provides procedures for performing specific migration tasks when moving a CMVC from one version of the operating system or DBMS or CMVC, to another version.

Please follow the recommendations described in “Overall recommendations” on page 5.

HOW TO HAVE TWO DIFFERENT CMVC VERSIONS AT ONCE

It is possible to have two different versions of CMVC (for example 2.3.0.25 and 2.3.1.4) in the same host, provided that the following conditions are met. This assumes that 2.3.0 is already installed and you want to install 2.3.1 later on, without overwriting 2.3.0.

If you want to have 2.3.0.14 and 2.3.0.25 installed at the same time for example, then in this document 2.3.0 will refer to 2.3.0.14 and 2.3.1 will refer to the newest one, 2.3.0.25.

1. When installing CMVC, use different home directories, such as:

   ```
   CMVC 2.3.0:    /usr/lpp/cmvc
   CMVC 2.3.1:    /usr/lpp/cmvc231
   ```

   This is only possible when using "tar images", which are available in all platforms. It is not possible when using "installp images" (only for AIX).

2. Specify a different directory for the message catalogs for CMVC 2.3.1, such as:

   ```
   CMVC 2.3.0:    /usr/lib/nls/msg/En_US
   CMVC 2.3.1:    /usr/lpp/cmvc231/nls/msg/En_US
   ```

   Just in case, before installing CMVC 2.3.1, make a copy of the 2.3.0 message catalog files from their system location which is /usr/lib/nls/msg/En_US to /usr/lpp/cmvc/nls/msg/En_US (for 2.3.0).

3. Change the NLSPATH for the CMVC 2.3.1 families to reflect the new location, and assuming that LANG=En_US for AIX or LANG=C for HP-UX and Solaris:

   ```
   CMVC 2.3.0:  NLSPATH=/usr/lib/nls/msg/%L/%N
   CMVC 2.3.1:  NLSPATH=/usr/lpp/cmvc231/%L/%N
   ```

4. The bitmaps used by the GUI were not changed at all between CMVC 2.3.0 and 2.3.1. They are located in /usr/include/X11/bitmaps. That is, there is no problem is the 2.3.0 ones are overwritten by 2.3.1.

5. The default X11 resources are the same for both 2.3.0 and 2.3.1. They are located in /usr/lib/X11/app-defaults. That is, there is no problem is the 2.3.0 ones are overwritten by 2.3.1.

AIX 4: NECESSARY LINKS IF USING THE EN_US LOCALE

If you are going to use the "en_US" locale in AIX 4 (which is the default), then it is suggested to do one of the following because the only locale provided by CMVC is En_US:

- Create symbolic links.

  1. Login as root.
  2. Do the following symbolic links:

     ```
     ln -s /usr/lib/nls/msg/En_US/cmvc.cat     /usr/lib/nls/msg/en_US/cmvc.cat
     ln -s /usr/lib/nls/msg/En_US/cmvchelp.cat /usr/lib/nls/msg/en_US/cmvchelp.cat
     ln -s /usr/lib/nls/msg/En_US/cmvcui.cat   /usr/lib/nls/msg/en_US/cmvcui.cat
     ```

- Do not use %L in the NLSPATH environment variable, instead explicitly use "En_US" in the .profile:

  ```
  export NLSPATH=/usr/lib/nls/msg/En_US/%N
  ```

- In the worst case scenario do:

  ```
  login as the CMVC family id
  cd $HOME
  mkdir -p nls/msg/En_US
  cp /usr/lib/nls/msg/En_US/cmvc*.cat nls/msg/msg/En_US/.
  export NLSPATH=$HOME/nls/msg/En_US/%N
  ```

  Modify the NLSPATH in the .profile.

MIGRATING A CMVC FAMILY FROM DB2 V1 TO DB2 UDB V5

This section describes how to migrate from CMVC 2.2 with DB2 V1 on AIX 3.2.5 to CMVC 2.3.1.4 with DB2 UDB V5 on AIX 4.2.1.

It will be highly advisable to keep for a short period after migration the current Host A with CMVC 2.2 (which is out of service) with DB2 V1 on AIX 3.2.5. Only when you are satisfied with the operational characteristics of the new migrated family in Host B, you can delete the family in Host A.

The recommended migration sequence is based on “Cloning/migration of a CMVC family from one host to another” on page 25, “Migration tips when moving from one version of CMVC to another” on page 31 and “Migrating to CMVC 2.3.1 (which supports the Year 2000)” on page 19. However, some important details are highlighted below:

1. Add a CMVC host list entry for the target Host B (AIX 4).

2. In the source Host A (AIX 3.2.5) upgrade CMVC 2.2 on DB2 V1 to CMVC 2.3.0.25 on DB2 V1. See the Appendix B of the CMVC Server 2.3 manual for details.
3. In the source Host A (AIX 3.2.5) migrate DB2 V1 to DB2 V2 on AIX 3.2.5. This should not impact CMVC. However, you should look at the DB2 documentation for migrating the database.
4. At this stage you will have CMVC 2.3.0.25, DB2 V2 on AIX 3.2.5. For the details on migrating from DB2 V2 to DB2 UDB V5 see “Migrating a CMVC family from DB2 V2 to DB2 UDB V5”

MIGRATING A CMVC FAMILY FROM DB2 V2 TO DB2 UDB V5

If you are currently using CMVC under DB2 V2 on AIX4, and if you would like to migrate the CMVC family to VisualAge TeamConnection which uses DB2 Universal Database Version 5 (DB2 UDB V5), then it is recommended that first you migrate your CMVC family from DB2 V2 to DB2 UDB V5, in that way you can have only one version of DB2 in the host.

- To migrate from DB2 V2 to DB2 UDB V5 in the same host do the following:

  1. Backup the DB2 V2 database of the CMVC family.
  2. Uninstall DB2 V2.
  3. Install DB2 UDB V5.

     You can follow the instructions provided in the technical report mentioned in “Other related technical reports” on page 7.
  4. Follow all the instructions from Chapter 6 "Migrating from Previous Versions", of the DB2 UDB V5 Quick Beginnings for Unix manual.

- To migrate a CMVC family from DB2 V2 in one host to DB2 UDB V5 in another host, do the following:

  1. Follow the steps for migrating a CMVC family from one host to another. See “Cloning/migration of a CMVC family from one host to another” on page 25.
  2. Restore the backup file for the CMVC family that used DB2 V2.

     After the restore you will see the SQL2517W warning message confirming that the database was migrated to the current release.
  3. If migrating from a CMVC 2.3.0 family, then follow the steps for upgrading a CMVC family to 2.3.1. See “Migrating to CMVC 2.3.1 (which supports the Year 2000)” on page 19.

MIGRATION FROM INFORMIX 5 TO DB2 V2 IN AIX 3 TO DB2 UDB V5 IN AIX 4

Here is how we recommend making the migration from CMVC 2.3.0 using Informix 5 on AIX 3.2 to CMVC 2.3.0 using DB2 V2 on AIX 3.2.5 to the final stage of CMVC 2.3.1.4 on DB2 UDB V5 on AIX 4.

As in previous examples, we will call the current machine Host A and the new machine Host B.

1. Perform a complete backup of the current CMVC family database (under Informix) and file system on Host A. See the Informix documentation for specific directions. For more details see “How to make a complete backup of your CMVC family” on page 9.
2. Install DB2 V2 on AIX 3.2.5 and then migrate the database from Informix to DB2 in Host A as explained in the CMVC Server manual, Appendix D.

   However, do not migrate CMVC or AIX on Host A yet.
3. Verify that the CMVC family is working correctly under DB2 V2 on Host A.
4. Make a complete backup of the CMVC family database (under DB2 V2) and file system on Host A.
5. Migrate from Host A running CMVC 2.3.0 on DB2 V2 on AIX 3.2.5 to Host B running CMVC 2.3.1 on DB2 UDB V5 on AIX 4.x. For more details see “Cloning/migration of a CMVC family from one host to another” on page 25.
6. Apply dbSetDate to update the CMVC 2.3.0 database to 2.3.1. For more details see “Migrating to CMVC 2.3.1 (which supports the Year 2000)” on page 19.
7. Verify that the CMVC family is working correctly under DB2 UDB V5 on Host B.
8. Make a complete backup of the CMVC family database (under DB2 UDB V5) and file system on Host B (now under AIX 4).

MIGRATION FROM INFORMIX 5 IN AIX 3 TO DB2 UDB V5 IN AIX 4

Here is how we recommend making the migration from CMVC 2.3.0 using Informix 5 on AIX 3.2 to CMVC 2.3.1.4 on DB2 UDB V5 on AIX 4.

As in previous examples, we will call the current machine Host A and the new machine Host B.

1. Perform a complete backup of the current CMVC family database (under Informix) and file system on Host A. See the Informix documentation for specific directions. For more details see “How to make a complete backup of your CMVC family” on page 9.

2. Install DB2 UDB on AIX 4 and then migrate the database from Informix 5 in Host A to DB2 UDB in Host B as explained below:
3. Perform the migration steps specified in “Migration from Informix 7 to DB2 UDB V5 in AIX 4.”
4. Apply dbSetDate to update the CMVC 2.3.0 database to 2.3.1 in Host B. For more details see “Migrating to CMVC 2.3.1 (which supports the Year 2000)” on page 19.
5. Verify that the CMVC family is working correctly under DB2 UDB V5 on Host B.
6. Make a complete backup of the CMVC family database (under DB2 UDB V5) and file system on Host B (now under AIX 4).

MIGRATION FROM INFORMIX 7 TO DB2 UDB V5 IN AIX 4

This chapter describes the migration from CMVC 2.3.0 using Informix 7 on AIX 4.x or HP-UX 10.x to CMVC 2.3.1 using DB2 UDB V5 on AIX 4.

Step 1: Pre-migration tasks

This section provides a summary of the pre-migration tasks to be done in both hosts. For more details see “Pre-migration steps to be done in source Host A” on page 26 and “Migration steps to be done in target Host B” on page 26.

Pre-migration tasks to be done in the source Host A

In the source Host A (AIX 4.x or HP-UX 10.x) do the following:

1. Login as the CMVC family user ID.
2. Stop the family daemons:

   ```
   $ stopCMVC $CMVC_FAMILY
   ```
3. Make a tar file of the $HOME directory for the CMVC family:

   ```
   $ cd $HOME
   $ tar -cvf family.tar .
   ```

Pre-migration tasks to be done in the target Host B

In the target Host B (AIX 4) do the following:

1. Login as root.
2. Install DB2 UDB V5 and create a DB2 instance (default: db2inst1). See the installation

instructions from the technical report 29.3076 "Configuration and Administration of DB2 Universal Database V5 by users of VisualAge TeamConnection V3".

3. Install CMVC 2.3.1.x for DB2 UDB V5.
4. Change to the /usr/lpp/cmvc directory.
5. The migtools.tar.Z file is included in CMVC 2.3.1.5 in /usr/lpp/cmvc/samples. You will need to uncompress it and untar it.
6. If you do not have CMVC 2.3.1.5, then proceed with this section. From the CMVC ftp site:
   a. cd ps/products/cmvc
   b. cd doc/tr
   c. binary
   d. get migtools.tar.Z
7. Uncompress and untar the file migtools.tar.Z:

   ```
   $ uncompress migtools.tar.Z
   $ tar -xvf migtools.tar
   ```

   Some of the extracted files are shown below:

   ```
   db2Convert/db2InsertRemarks
   db2Convert/db2InsertRemarks.bnd
   db2Convert/infRemsEor
   db2Convert/note1.bnd
   db2Convert/version1.bnd
   db2Convert/*dumptbls
   db2Convert/*dumptblsrems
   ```

   All the extracted files MUST be located in the db2Convert directory. DO NOT copy the new *.bnd files into the /usr/lpp/cmvc/bind directory.
8. Create the user id for the CMVC family; the group id must be the same as the group id from the DB2 instance. It is recommended to use the same name as the old CMVC family. If possible try to use the same actual user number.
9. Update /etc/hosts and /etc/services with the family name and port number. It is recommended to use the same data as the old CMVC family.
10. Login as the CMVC user ID.
11. Modify the .profile.
12. Logout and login again to have a clean environment.
13. Get the family.tar file obtained in “Pre-migration tasks to be done in the source Host A” on page 41, and put it in $HOME.
14. Untar the file:

    ```
    $ tar -xvf family.tar
    ```

15. Ensure that the file permissions and ownership are correct for the new files that were just expanded into the new CMVC family.

    ```
    find . -exec chown userId:groupId {} \;
    ```

    You need to specify the proper userId and groupId, for example:

    ```
    $ find . -exec chown cmpc3mig:db2inst1 {} \;
    ```

16. Do not issue "mkfamily". By using the "tar -xvf" command on the tar file you recreated the directories and files needed for the family.
17. Create the DB2 database for the CMVC family by using:

    ```
    $ mkdb
    ```

    Do not use the -d option for the mkdb command as the command needs to read the $HOME/configField files to ensure that the Defects, Files and Users tables are created with the appropriate configurable fields.

Up to this point you have recreated the family files and created a new (almost empty) database. The next steps will tell you how to populate it with the data from the old database.

Step 2: Process most of the tables (except Versions and Notes)

The following instructions were obtained from the CMVC Server manual, version 2.3, Appendix D "Migrating to CMVC Server V2.3.0 for DB2". These instructions apply also for CMVC 2.3.1.

The database consists of several tables. All these tables, except for Versions and Notes, can be migrated at the same time by following the procedure below.

In separate steps you can migrate the tables for Versions and Notes. See “Step 3: Process Versions and Notes” on page 38 and “Step 4: Post-migration steps in target Host B” on page 40.

Dumping tables in source Host A

In the source Host A do the following:

1. Login as the CMVC family user ID.
2. Create a directory to store the flat files with the data extracted from the tables of the database:

   ```
   $ mkdir $HOME/flatfiles
   ```

3. Set the environment FLATFILEDIR to this new directory:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

4. Dump the tables (except Notes and Versions) to flat files:

   ```
   $ dbexport -o $FLATFILEDIR  $CMVC_FAMILY
   ```

5. The above tools will create temporary files in /tmp. Just in case these files contain warnings or errors, take a look at them.

6. Take a look at the files in $FLATFILEDIR, which contain the extracted data. These files have 1 row per object, and the fields for the object are separated by a vertical bar "|".

   It might be possible that some fields such as in the abstract of Defects may contain "|"; in this case you need to replace these instances to another character, such as "!" in order to avoid problems when parsing the data during the import phase later on; that is, a field might be truncated because it found a "|" that is not a field separator.

7. Make a tar file of the flat files:

   ```
   $ cd $HOME
   $ tar -cvf flatfiles.tar $FLATFILEDIR
   ```

Loading most of the tables in target Host B

In the target Host B do the following:

1. Login as the CMVC family user ID.

2. Create a directory to store the flat files with the data extracted from the tables of the database:

   ```
   $ mkdir $HOME/flatfiles
   ```

3. Set the environment FLATFILEDIR to this new directory:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

4. Get the flatfiles.tar file and untar it:

   ```
   $ tar -xvf flatfiles.tar
   ```

5. Load (insert/import) the tables:

   ```
   $ $HOME/db2Convert/infx7db2tbls
   ```

6. Make a backup of your database now.

Step 3: Process Versions and Notes

Dumping Versions and Notes in source Host A

The dumping of the Versions and Notes tables was already accomplished in “Dumping tables in source Host A” on page 37.

Loading Versions and Notes in target Host B

In the target Host B do the following:

1. You need to bind these temporary bind files. You will rebind the original, permanent bind files again at the end of the process.
   a. $ db2 connect to $LOGNAME
   b. $ db2 bind /usr/lpp/cmvc/samples/db2Convert/db2InsertRemarks.bnd
   c. $ db2 bind /usr/lpp/cmvc/samples/db2Convert/note1.bnd
   d. $ db2 bind /usr/lpp/cmvc/samples/db2Convert/version1.bnd
2. Specify the directory where the "flat files" are located, such as:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

3. Copy the exported files from the source Host A into $HOME/flatfiles.

   Because these tables may contain remarks that are split between several lines and because they may contain vertical bars ('|'), then it is necessaty to process the exported files to identify the end of each record with the sequence !_!. This avoids the problem of embedded | vertical bars and line feeds \n inside the comments for Notes and Versions.

4. Execute the following from your Informix CMVC family id:

   ```
   $ cd $FLATFILEDIR
   $ ../db2Convert/infRemsEor N notes00119.unl
   $ ../db2Convert/infRemsEor V versi00128.unl
   ```

   Ensure that the proper names for the Notes and Versions are specified, in case that they are different as the ones used in the above example.

5. You will get 2 files with extension of ".cmvc", such as:

   ```
   notes00119.unl.cmvc
   versi00128.unl.cmvc
   ```

   The format is slightly different than the ones that have the suffix of *.unl, in which the end of a database record is represented by the 3 characters !_!.

   Also, the combination of back slash + vertical bar (\|) is replaced by back slash + exclamation (\!) to avoid problems with the parsing by db2InsertRemarks.

6. Edit the $FLATFILEDIR/versi00128.unl.cmvc file and delete the first entry that contains all 0's.

If you do not do this, then you will get an error when trying to import it, because it will be a duplicate one. The symptom is error: 0010-061 Database error, -803

7. Insert the Notes:

   ```
   $ db2InsertRemarks  n  $FLATFILEDIR/notes00119.unl.cmvc
   ```

8. Insert the Versions:

   ```
   $ db2InsertRemarks  v  $FLATFILEDIR/versi00128.unl.cmvc
   ```

9. If there were remarks in the Versions file that contained one or more |, these rows were not processed. These rows were placed in $FLATFILEDIR/Versions.cmvc.aux. You will need to edit this file to remove any | (vertical bars) that are located in the remarks field. This field is usually located between the 7th and the 8th vertical bar. The changeDate field is located between the 8th and the 9th vertical bar. You will need to replace any extra vertical bars manually, by changing it from | to !, for example.

10. After you edit the $FLATFILEDIR/Versions.cmvc.aux, you can insert the remaining versions into the database:

    ```
    $ db2InsertRemarks  v  $FLATFILEDIR/Versions.cmvc.aux
    ```

11. Rebind again with the original, permanent bind files:

    ```
    $ db2bind $LOGNAME
    ```

Step 4: Post-migration steps in target Host B

Perform the tasks described in “Post-migration steps to be done in target Host B” on page 28.

MIGRATION FROM CMVC 2.3.0 SYBASE TO CMVC 2.3.1 DB2 UDB V5

This chapter describes the migration from CMVC 2.3.0 using Sybase 4.9/Sybase 11 on AIX 3.2.5/Solaris to CMVC 2.3.1 using DB2 UDB V5 on AIX 4.

Step 1: Pre-migration tasks

This section provides a summary of the pre-migration tasks to be done in both hosts. For more details see “Pre-migration steps to be done in source Host A” on page 26 and “Migration steps to be done in target Host B” on page 26.

Pre-migration tasks to be done in the source Host A

In the source Host A (AIX 3.2.5 or Solaris) do the following:

1. Login as the CMVC family user ID.
2. Stop the family daemons:

   ```
   $ stopCMVC $CMVC_FAMILY
   ```

3. Make a tar file of the $HOME directory for the CMVC family:

   ```
   $ cd $HOME
   $ tar -cvf family.tar .
   ```

Pre-migration tasks to be done in the target Host B

In the target Host B (AIX 4) do the following:

1. Login as root.
2. Install DB2 UDB V5 and create a DB2 instance (default: db2inst1). See the installation instructions from the technical report 29.3076 "Configuration and Administration of DB2 Universal Database V5 by users of VisualAge TeamConnection V3".
3. Install CMVC 2.3.1.x for DB2 UDB V5.
4. Change to the /usr/lpp/cmvc directory.
5. The migtools.tar.Z file is included in CMVC 2.3.1.5 in /usr/lpp/cmvc/samples.
6. If you do not have CMVC 2.3.1.5, then proceed with this section. From the CMVC ftp site:
   a. cd ps/products/cmvc
   b. cd doc/tr
   c. binary
   d. get migtools.tar.Z
7. Uncompress and untar the file:

   ```
   $ uncompress migtools.tar.Z
   $ tar -xvf migtools.tar
   ```

   Some of the extracted files are shown below:

   ```
   db2Convert/db2InsertRemarks
   db2Convert/db2InsertRemarks.bnd
   db2Convert/note1.bnd
   db2Convert/version1.bnd
   db2Convert/*dumptbls
   db2Convert/*dumptblsrems
   ```

   All the extracted files MUST be located in the db2Convert directory. DO NOT copy the new *.bnd files into the /usr/lpp/cmvc/bind directory.

8. Create the user id for the CMVC family; the group id must be the same as the group id from the DB2 instance. It is recommended to use the same name as the old CMVC family. If possible try to use the same actual user number.
9. Update /etc/hosts and /etc/services with the family name and port number. It is recommended to use the same data as the old CMVC family.
10. Login as the CMVC user ID.
11. Modify the .profile.
12. Logout and login again to have a clean environment.
13. Get the family.tar file obtained in “Pre-migration tasks to be done in the source Host A” on page 41, and put it in $HOME.
14. Untar the file:

    ```
    $ tar -xvf family.tar
    ```
15. Ensure that the file permissions and ownership are correct for the new files that were just expanded into the new CMVC family.

    ```
    find . -exec chown userId:groupId {} \;
    ```

    You need to specify the proper userId and groupId, for example:

    ```
    $ find . -exec chown cmpc3mig:db2inst1 {} \;
    ```
16. Do not issue "mkfamily". By using the "tar -xvf" command on the tar file you recreated the directories and files needed for the family.
17. Create the DB2 database for the CMVC family by using:

    ```
    $ mkdb
    ```

    Do not use the -d option for the mkdb command as the command needs to read the $HOME/configField files to ensure that the Defects, Files and Users tables are created with the appropriate configurable fields.

Up to this point you have recreated the family files and created a new (almost empty) database. The next steps will tell you how to populate it with the data from the old database.

Step 2: Process most of the tables (except Versions and Notes)

The following instructions were obtained from the CMVC Server manual, version 2.3, Appendix D "Migrating to CMVC Server V2.3.0 for DB2". These instructions apply also for CMVC 2.3.1.

The database consists of several tables. All these tables, except for Versions and Notes, can be migrated at the same time by following the procedure below.

In separate steps you can migrate the tables for Versions and Notes. See “Step 3: Process Versions and Notes” on page 44 and “Step 4: Post-migration steps in target Host B” on page 45.

Dumping tables in source Host A

In the source Host A do the following:

1. Login as the CMVC family user ID.
2. Create a directory to store the flat files with the data extracted from the tables of the database:

   ```
   $ mkdir $HOME/flatfiles
   ```

3. Set the environment FLATFILEDIR to this new directory:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

4. Dump the tables (except Notes and Versions) to flat files.
   - If using Informix, do (use the CMVC family name):

     ```
     $ dbexport -o $FLATFILEDIR  $CMVC_FAMILY
     ```

   - If using Oracle, ensure that $ORACLE_PASS is properly set with the password of the CMVC family user id and then issue:

     ```
     $ ora7dumptbls
     ```

   - If using Sybase, ensure that $SYBASE_PASS is properly set with the password of the CMVC family user id and then issue:

     ```
     $ sybdumptbls
     ```

5. The above tools will create temporary files in /tmp. Just in case these files contain warnings or errors, take a look at them.
6. Take a look at the files in $FLATFILEDIR, which contain the extracted data. These files have 1 row per object, and the fields for the object are separated by a vertical bar "|".

   It might be possible that some fields such as in the abstract of Defects may contain "|"; in this case you need to replace these instances to another character, such as "!" in order to avoid problems when parsing the data during the import phase later on; that is, a field might be truncated because it found a "|" that is not a field separator.

7. Make a tar file of the flat files:

   ```
   $ cd $HOME
   $ tar -cvf flatfiles.tar $FLATFILEDIR
   ```

Loading most of the tables in target Host B

In the target Host B do the following:

1. Login as the CMVC family user ID.
2. Create a directory to store the flat files with the data extracted from the tables of the database:

   ```
   $ mkdir $HOME/flatfiles
   ```

3. Set the environment FLATFILEDIR to this new directory:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

4. Get the flatfiles.tar file and untar it:

   ```
   $ tar -xvf flatfiles.tar
   ```

5. Load (insert/import) the tables:
   - If the files were obtained from a previous Sybase CMVC family, use:

     ```
     $ syb2db2tbls
     ```

   - If the files were obtained from a previous Oracle CMVC family, use:

     ```
     $ ora72db2tbls
     ```

   - If the files were obtained from a previous Informix CMVC family, use:

     ```
     $ infx2db2tbls
     ```

6. Make a backup of your database now.

Step 3: Process Versions and Notes

Dumping Versions and Notes in source Host A

In the source Host A do the following:

1. Execute the following file from your Sybase CMVC family id:

   ```
   $ sybdumptblsrems
   ```

   You will get 2 files with extension of ".cmvc".

   The format is slightly different than the ones that have the suffix of *.unl, in which the end of a database row is represented by the 3 characters !_!. This avoids the problem of embedded | vertical bars and line feeds \n inside the comments for Notes and Versions.

Loading Versions and Notes in target Host B

In the target Host B do the following:

1. You need to bind these temporary bind files. You will rebind the original, permanent bind files again at the end of the process.
   a. $ db2 connect to family
   b. $ db2 bind /usr/lpp/cmvc/db2Convert/db2InsertRemarks.bnd
   c. $ db2 bind /usr/lpp/cmvc/db2Convert/note1.bnd
   d. $ db2 bind /usr/lpp/cmvc/db2Convert/version1.bnd
2. Specify the directory where the "flat files" are located, such as:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

3. Copy the exported files from the source Host A into $HOME/flatfiles.
4. Edit the $HOME/flatfiles/Versions.cmvc file and delete the first entry that contains all 0's.

   If you do not do this, then you will get an error when trying to import it, because it will be a duplicate one. The symptom is error: 0010-061 Database error, -803
5. Insert the Notes:

   ```
   $ db2InsertRemarks  n  $FLATFILEDIR/Notes.cmvc
   ```

6. Insert the Versions:

   ```
   $ db2InsertRemarks  v  $FLATFILEDIR/Versions.cmvc
   ```

7. If there were remarks in the Versions file that contained one or more |, these rows were not processed. These rows were placed in $FLATFILEDIR/Versions.cmvc.aux. You will need to edit this file to remove any | (vertical bars) that are located in the remarks field. This field is usually located between the 7th and the 8th vertical bar. The changeDate field is located between the 8th and the 9th vertical bar. You will need to replace any extra vertical bars manually, by changing it from | to #, for example.
8. After you edit the $FLATFILEDIR/Versions.cmvc.aux, you can insert the remaining versions into the database:

   ```
   $ db2InsertRemarks  v  $FLATFILEDIR/Versions.cmvc.aux
   ```

9. Rebind again with the original, permanent bind files:

   ```
   $ db2bind $CMVC_FAMILY
   ```

Step 4: Post-migration steps in target Host B

Perform the tasks described in “Post-migration steps to be done in target Host B” on page 28.

MIGRATION FROM CMVC 2.3.0 SYBASE 4.9 TO CMVC 2.3.1 SYBASE 11

This chapter describes the migration from CMVC 2.3.0 using Sybase 4.9 on AIX 3.2.5/Solaris to CMVC 2.3.1 using Sybase 11 on Solaris.

Unfortunately, a backup (dump) file of a Sybase 4.9 cannot be restored (loaded) into a Sybase 11 database. Therefore, the bulk copy (bcp) utility from Sybase is used to dump the tables from the Sybase 4.9 database and to load the tables into the Sybase 11 database.

Step 1: Pre-migration tasks

This section provides a summary of the pre-migration tasks to be done in both hosts. For more details see “Pre-migration steps to be done in source Host A” on page 26 and “Migration steps to be done in target Host B” on page 26.

Pre-migration tasks to be done in the source Host A

In the source Host A (AIX 3.2.5 or Solaris) do the following:

1. Login as the CMVC family user ID.
2. Stop the family daemons:

   ```
   $ stopCMVC $CMVC_FAMILY
   ```

3. Make a tar file of the $HOME directory for the CMVC family:

   ```
   $ cd $HOME
   $ tar -cvf family.tar .
   ```

Pre-migration tasks to be done in the target Host B

In the target Host B (Solaris) do the following:

1. Login as root.
2. Install and configure Sybase 11.
3. Install CMVC 2.3.1.x for Sybase.
4. Change to the /usr/lpp/cmvc directory.
5. The migtools.tar.Z file is included in CMVC 2.3.1.5 in /usr/lpp/cmvc/samples.
6. If you do not have CMVC 2.3.1.5, then proceed with this section. From the CMVC ftp site:

   a. cd ps/products/cmvc

b. cd doc/tr

c. binary

d. get migtools.tar.Z

7. Uncompress and untar the file:

```
$ uncompress migtools.tar.Z
$ tar -xvf migtools.tar
```

Some of the extracted files are shown below:

```
db2Convert/sybdumptbls
db2Convert/sybdumptblsrems
db2Convert/sybloadtbls
db2Convert/sybloadtblsrems
```

All the extracted files MUST be located in the db2Convert directory. DO NOT copy the new *.bnd files into the /usr/lpp/cmvc/bind directory.

8. Create the user id for the CMVC family. It is recommended to use the same name as the old CMVC family. If possible try to use the same actual user number.
9. Update /etc/hosts and /etc/services with the family name and port number. It is recommended to use the same data as the old CMVC family.
10. Login as the CMVC user ID.
11. Modify the .profile.
12. Logout and login again to have a clean environment.
13. Get the family.tar file obtained in “Pre-migration tasks to be done in the source Host A” on page 46, and put it in $HOME.
14. Untar the file:

```
$ tar -xvf family.tar
```

15. Ensure that the file permissions and ownership are correct for the new files that were just expanded into the new CMVC family.

```
$ find . -exec chown userId:groupId {} \;
```

You need to specify the proper userId and groupId, for example:

```
$ find . -exec chown cmpc3mig:db2inst1 {} \;
```

16. Do not issue "mkfamily". By using the "tar -xvf" command on the tar file you recreated the directories and files needed for the family.
17. Create the database for the CMVC family by using:

```
$ mkdb
```

Do not use the -d option for the mkdb command as the command needs to read the $HOME/configField files to ensure that the Defects, Files and Users tables are created with the appropriate configurable fields.

Up to this point you have recreated the family files and created a new (almost empty) database. The next steps will tell you how to populate it with the data from the old database.

Step 2: Process most of the tables (except Versions and Notes)

The following instructions were obtained from the CMVC Server manual, version 2.3, Appendix D "Migrating to CMVC Server V2.3.0 for DB2". These instructions apply also for CMVC 2.3.1.

The database consists of several tables. All these tables, except for Versions and Notes, can be migrated at the same time by following the procedure below.

In separate steps you can migrate the tables for Versions and Notes. See “Step 3: Process Versions and Notes” on page 49 and “Step 4: Post-migration steps in target Host B” on page 51.

Dumping tables in source Host A

In the source Host A do the following:

1. Login as the CMVC family user ID.
2. Create a directory to store the flat files with the data extracted from the tables of the database:

   ```
   $ mkdir $HOME/flatfiles
   ```

3. Set the environment FLATFILEDIR to this new directory:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

4. Dump the tables (except Notes and Versions) to flat files. Because we are using Sybase, ensure that $SYBASE_PASS is properly set with the password of the CMVC family user id and then issue:

   ```
   $ ./db2Convert/sybdumptbls
   ```

   The tool will create temporary files in /tmp. Just in case these files contain warnings or errors, take a look at them.

5. Take a look at the files in $FLATFILEDIR, which contain the extracted data. These files have 1 row per object, and the fields for the object are separated by a vertical bar "|".

It might be possible that some fields such as in the abstract of Defects may contain "|"; in this case you need to replace these instances to another character, such as "!" in order to avoid problems when parsing the data during the import phase later on; that is, a field might be truncated because it found a "|" that is not a field separator.

6. Make a tar file of the flat files:

```
$ cd $HOME
$ tar -cvf flatfiles.tar $FLATFILEDIR
```

Loading most of the tables in target Host B

In the target Host B do the following:

1. Login as the CMVC family user ID.
2. Create a directory to store the flat files with the data extracted from the tables of the database:

```
$ mkdir $HOME/flatfiles
```

3. Set the environment FLATFILEDIR to this new directory:

```
$ export FLATFILEDIR=$HOME/flatfiles
```

4. Get the flatfiles.tar file and untar it:

```
$ tar -xvf flatfiles.tar
```

5. Edit the following files in $FLATFILEDIR:
   - CompMembers.unl: delete the first entry (2|2|0|1).
   - Components.unl: delete the first entry (for component "root").
   - Levels.unl: delete the first entry (0|||||).
   - Releases.unl: delete the first entry (0||0||).
   - Users.unl: delete the first TWO entries.
6. Cleanup the current Sequence table inside the database:

```
$ isql -P$SYBASE_PASS
1> delete from Sequence where name in ('defect','source','general')
2> go
```

7. Load (insert/import) the tables. Because the files were obtained from a previous Sybase CMVC family, use:

```
$ ./db2Convert/sybloadtbls
```

8. Make a backup of your database now.

Step 3: Process Versions and Notes

Dumping Versions and Notes in source Host A

In the source Host A do the following:

1. Execute the following file from your Sybase CMVC family id:

   ```
   $ ./db2Convert/sybdumptblsrems
   ```

   You will get 2 files with extension of ".cmvc".

   The format is slightly different than the ones that have the suffix of *.unl, in which the end of a database row is represented by the 3 characters !_!. This avoids the problem of embedded | vertical bars and line feeds \n inside the comments for Notes and Versions.

Loading Versions and Notes in target Host B

In the target Host B do the following:

1. Specify the directory where the "flat files" are located, such as:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

2. Copy the exported files from the source Host A into $HOME/flatfiles.
3. Edit the $HOME/flatfiles/Versions.cmvc file and delete the first entry that contains all 0's.

   If you do not do this, then you will get an error when trying to import it, because it will be a duplicate one.

4. Insert the Notes and Versions:

   ```
   $ ./db2Convert/sybloadtblsrems
   ```

5. If there were remarks in the Versions file that contained one or more |, these rows were not processed. These rows were placed in $FLATFILEDIR/Versions.cmvc.aux. You will need to edit this file to remove any | (vertical bars) that are located in the remarks field. This field is usually located between the 7th and the 8th vertical bar. The changeDate field is located between the 8th and the 9th vertical bar. You will need to replace any extra vertical bars manually, by changing it from | to #, for example.
6. Rename the original $FLATFILEDIR/Versions.cmvc or Notes.cmvc to something else.
7. After you edit the $FLATFILEDIR/Versions.cmvc.aux, rename it to $FLATFILEDIR/Versions.cmvc You can insert the remaining versions into the database by entering the following command and specifying Notes or Versions:

   ```
   $ ./db2Convert/sybloadtblsrems
   ```

Step 4: Post-migration steps in target Host B

Perform the tasks described in “Post-migration steps to be done in target Host B” on page 28.

Step 5: Migrate the family in target Host B to CMVC 2.3.1

Perform the tasks described in “Migrating to CMVC 2.3.1 (which supports the Year 2000)” on page 19.

MIGRATION FROM CMVC 2.3.0 ORACLE 7 TO CMVC 2.3.1 DB2 UDB V5

This chapter describes the migration from CMVC 2.3.0 using Oracle 7 (from AIX, HP-UX or Solaris) to CMVC 2.3.1 using DB2 UDB V5 on AIX 4.

Step 1: Pre-migration tasks

This section provides a summary of the pre-migration tasks to be done in both hosts.

1. You need to perform the steps mentioned in: “Pre-migration steps to be done in source Host A” on page 26.
2. You need to perform the steps mentioned in: “Migration steps to be done in target Host B” on page 26.

Pre-migration tasks to be done in the source Host A

In the source Host A (AIX 4.x, HP-UX 10.x or Solaris) do the following:

1. Login as the CMVC family user ID.
2. Stop the family daemons:

   ```
   $ stopCMVC $CMVC_FAMILY
   ```

3. Make a tar file of the $HOME directory for the CMVC family. You will need to specify a directory where there is enough file system space to create the tar file, such as /tmp in this example:

   ```
   $ cd $HOME
   $ tar -cvf /tmp/family.tar .
   ```

4.
5. Change to the /usr/lpp/cmvc directory.

6. From the CMVC ftp site:

   a. cd ps/products/cmvc
   b. cd doc/tr
   c. binary
   d. get migtools.tar.Z

7. Uncompress and untar the file migtools.tar.Z. This assumes that you have changed the directory to /usr/lpp/cmvc:

   ```
   $ uncompress migtools.tar.Z
   $ tar -xvf migtools.tar
   ```

   Some of the extracted files are shown below:

   ```
   db2Convert/db2InsertRemarks
   db2Convert/db2InsertRemarks.bnd
   db2Convert/note1.bnd
   db2Convert/version1.bnd
   db2Convert/*dumptbls
   db2Convert/*dumptblsrems
   ```

   All the extracted files MUST be located in the /usr/lpp/cmvc/db2Convert directory. DO NOT copy the new *.bnd files into the /usr/lpp/cmvc/bind directory.

8. The latest versions of the ora7dump and ora7dumprems tools were developed under Oracle 7.3.4. Thus, in case that there are problems with these tools because you have Oracle 7.1 or 7.2, then you may need to perform the following step, in order to use the appropriate version of the tools:

   ```
   cp -p  ora7dump.71 ora7dump
   cp -p  ora7dumprems.71 ora7dumprems
   ```

Pre-migration tasks to be done in the target Host B

In the target Host B (AIX 4) do the following:

1. Login as root.
2. Install DB2 UDB V5 and create a DB2 instance (default: db2inst1). See the installation instructions from the technical report 29.3076 "Configuration and Administration of DB2 Universal Database V5 by users of VisualAge TeamConnection V3".
3. Install CMVC 2.3.1.x for DB2 UDB V5.
4. Change to the /usr/lpp/cmvc directory.
5. From the CMVC ftp site:

   a. cd ps/products/cmvc
   b. cd doc/tr
   c. binary

d. get migtools.tar.Z

6. Uncompress and untar the file migtools.tar.Z. This assumes that you have changed the directory to /usr/lpp/cmvc:

   ```
   $ uncompress migtools.tar.Z
   $ tar -xvf migtools.tar
   ```

   Some of the extracted files are shown below:

   ```
   db2Convert/db2InsertRemarks
   db2Convert/db2InsertRemarks.bnd
   db2Convert/note1.bnd
   db2Convert/version1.bnd
   db2Convert/*dumptbls
   db2Convert/*dumptblsrems
   ```

   All the extracted files MUST be located in the /usr/lpp/cmvc/db2Convert directory. DO NOT copy the new *.bnd files into the /usr/lpp/cmvc/bind directory.

7. Create the user id for the CMVC family; the group id must be the same as the group id from the DB2 instance. It is recommended to use the same name as the old CMVC family. If possible try to use the same actual user number.

8. Update /etc/hosts and /etc/services with the family name and port number. It is recommended to use the same data as the old CMVC family.

9. Login as the CMVC user ID.

10. Modify the .profile.

11. Logout and login again to have a clean environment.

12. Get the family.tar file obtained in “Pre-migration tasks to be done in the source Host A” on page 41, and put it in $HOME.

13. Untar the file:

    ```
    $ tar -xvf family.tar
    ```

14. You will need to update the .profile file in order to reflect the necessary environment variables for DB2. You can take a look at the following sample profile for more details:

    ```
    /usr/lpp/cmvc/install/profile.db2
    ```

15. Add the following path to PATH, in order to find the tools for migration:

    ```
    export PATH=$PATH:$CMVC_HOME/db2Convert
    ```

16. Ensure that the file permissions and ownership are correct for the new files that were just expanded into the new CMVC family.

    ```
    find . -exec chown userId:groupId {} \;
    ```

    You need to specify the proper userId and groupId, for example:

    ```
    $ find . -exec chown cmpc3mig:db2inst1 {} \;
    ```

17. Do not issue "mkfamily". By using the "tar -xvf" command on the tar file you recreated the directories and files needed for the family.
18. Create the DB2 database for the CMVC family by using:

    ```
    $ mkdb
    ```

    Do not use the -d option for the mkdb command as the command needs to read the $HOME/configField files to ensure that the Defects, Files and Users tables are created with the appropriate configurable fields.

Up to this point you have recreated the family files and created a new (almost empty) database. The next steps will tell you how to populate it with the data from the old database.

Step 2: Process most of the tables (except Versions and Notes)

The following instructions were obtained from the CMVC Server manual, version 2.3, Appendix D "Migrating to CMVC Server V2.3.0 for DB2". These instructions apply also for CMVC 2.3.1.

The database consists of several tables. All these tables, except for Versions and Notes, can be migrated at the same time by following the procedure below.

In separate steps you can migrate the tables for Versions and Notes. See “Step 3: Process Versions and Notes” on page 55 and “Step 4: Post-migration steps in target Host B” on page 57.

Dumping tables in source Host A

In the source Host A do the following:

1. Login as the CMVC family user ID.
2. Create a directory to store the flat files with the data extracted from the tables of the database:

   ```
   $ mkdir $HOME/flatfiles
   ```

3. Set the environment FLATFILEDIR to this new directory:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```

4. Dump the tables (except Notes and Versions) to flat files. Ensure that $ORACLE_PASS is properly set with the password of the CMVC family user id and then issue:

   ```
   $   $CMVC_HOME/db2Convert/ora7dumptbls
   ```

5. Dump the Notes and Versions tables to flat files. Ensure that $ORACLE_PASS is properly set with the password of the CMVC family user id and then issue:

```
$   $CMVC_HOME/db2Convert/ora7dumptblsrems
```

6. The above tools may create temporary files in /tmp. Just in case these files contain warnings or errors, take a look at them.
7. Take a look at the files in $FLATFILEDIR, which contain the extracted data. These files have 1 row per object, and the fields for the object are separated by a vertical bar "|".

   The records for the Notes and Versions files will be separated by the sequence: |_|

   It might be possible that some fields such as in the abstract of Defects may contain "|"; in this case you need to replace these instances to another character, such as "!" in order to avoid problems when parsing the data during the import phase later on; that is, a field might be truncated because it found a "|" that is not a field separator.
8. Make a tar file of the flat files:

```
$ cd $HOME
$ tar -cvf flatfiles.tar flatfiles
```

Loading most of the tables in target Host B

In the target Host B do the following:

1. Login as the CMVC family user ID.
2. Create a directory to store the flat files with the data extracted from the tables of the database:

```
$ mkdir $HOME/flatfiles
```

3. Set the environment FLATFILEDIR to this new directory:

```
$ export FLATFILEDIR=$HOME/flatfiles
```

4. Get the flatfiles.tar file and untar it:

```
$ tar -xvf flatfiles.tar
```

5. Load (insert/import) the tables:

```
$   $CMVC_HOME/db2Convert/ora72db2tbls
```

6. The above tool creates temporary files in /tmp. Just in case these files contain warnings or errors, take a look at them.
7. Make a backup of your database now.

Step 3: Process Versions and Notes

Dumping Versions and Notes in source Host A

The dumping of the Versions and Notes tables was already accomplished in “Dumping tables in source Host A” on page 54.

Loading Versions and Notes in target Host B

In the target Host B do the following:

1. You need to bind these temporary bind files. You will rebind the original, permanent bind files again at the end of the process.
   a. $ db2 connect to $LOGNAME
   b. $ db2 bind /usr/lpp/cmvc/db2Convert/db2InsertRemarks.bnd
   c. $ db2 bind /usr/lpp/cmvc/db2Convert/note1.bnd
   d. $ db2 bind /usr/lpp/cmvc/db2Convert/version1.bnd
2. Specify the directory where the "flat files" are located, such as:

   ```
   $ export FLATFILEDIR=$HOME/flatfiles
   ```
3. Edit the $FLATFILEDIR/Versions.unl file and delete the first entry that contains all 0's.

   If you do not do this, then you will get an error when trying to import it, because it will be a duplicate one.
4. Insert the Notes:

   ```
   $ db2InsertRemarks  n  $FLATFILEDIR/Notes.unl
   ```
5. Insert the Versions:

   ```
   $ db2InsertRemarks  v  $FLATFILEDIR/Versions.unl
   ```
6. If there were remarks in the Versions file that contained one or more |, these rows were not processed. These rows were placed in $FLATFILEDIR/Versions.unl.aux. You will need to edit this file to remove from the remarks field, any | (vertical bars) or to replace them with exclamation points. This field is located between the 7th and the 8th vertical bar. The changeDate field is located between the 8th and the 9th vertical bar; be careful to NOT change these other vertical bars that are true field separators.
7. After you edit the $FLATFILEDIR/Versions.unl.aux, you can insert the remaining versions into the database:

   ```
   $ db2InsertRemarks  v  $FLATFILEDIR/Versions.unl.aux
   ```
8. Rebind again with the original, permanent bind files:

   ```
   $ db2bind $LOGNAME
   ```

Step 4: Post-migration steps in target Host B

Perform the tasks described in “Post-migration steps to be done in target Host B” on page 28.

WARNING WHEN MIGRATING A DB2 FAMILY FROM AIX 3 TO AIX 4

Notes:

1. Because there are differences between AIX 3 and AIX 4, we make 2 versions of CMVC starting with 2.3.0.21, one for AIX 3 and another for AIX 4. However, for CMVC 2.3.1, we provide only the AIX 4 version.
2. We do NOT recommend to use CMVC 2.3.0.25 for AIX 3 on an AIX 4 system; thus, from our home page you can download the the desired AIX 4 code for CMVC 2.3.0.25 or 2.3.1.4.
3. The DB2 variable (DB2DBDFT) is needed to identify the default database to be used.

Changing the locale of a DB2 database (using only English characters)

If your source CMVC DB2 database uses the En_US locale (the default one in AIX 3), and if you are using only English characters (that is, no accented characters), then you can move the DB2 database as follows in order to change the locale to en_US (the default in AIX 4). This procedure is based on the technical report 29.3088 "Moving a VisualAge TeamConnection Version 3 Family":

1. Follow the instructions of the TR to export the data from the source CMVC DB2 database.
2. When creating the new target CMVC family, specify LANG=en_US and do not specify the DB2_CODESET and DB2_TERRITORY in order to pick up the default values.
3. Follow the instructions of the TR to import the data into the target CMVC DB2 database.

This TR is available from:

`ftp://ftp.software.ibm.com/ps/products/teamconnection/papers/trmovedb.pdf`

New DB2_CODESET and DB2_TERRITORY variables

The mkdb CMVC installation utility (which in turn calls the mkrmchdf shell script) and the sample profile.db2 were changed to use codeset/territory for DB2 during the creation of a DB2 database for the CMVC family. The new variables are DB2_CODESET and DB2_TERRITORY.

In common situations this is not really needed, but when migrating CMVC DB2 families from AIX 3 to AIX 4 it is critical that these variables are in sync: in AIX 3 the default is En_US.IBM-850 but in AIX 4 the new default is en_US.ISO8859-1.

If the user does not specify these variables, they will default to the proper value according to the version of AIX.

See the sample /usr/lpp/cmvc/install/profile.db2 for actual usage of these variables.

In short, if you have a family created in AIX 3 that used the defaults:

```
LANG=En_US
DB2_CODESET="IBM-850"
DB2_TERRITORY="En_US"
```

Then you need to install the locale En_US in AIX 4 (the output of "locale -a" should show En_US), and specify the above 3 variables when creating the database in AIX 4; do not use the default en_US locale in AIX 4 for that family.

You can use the following DB2 command to find out this data:

```
$ db2 get database configuration for $CMVC_FAMILY | head
```

The output would look like this (see the last 4 lines):

```
      Database Configuration for Database cmpc3db2

Database configuration release level                  = 0x0800
Database release level                                = 0x0800

Database territory                                    = en_US
Database code page                                    = 819
Database code set                                     = ISO8859-1
Database country code                                 = 1
```

If the En_US locale is not installed or it is not properly specified, then when doing the DB2 restore operation in AIX 4 the SQL 2548 error message will indicate the failure.

CMVC ORACLE 7.1 CODE CANNOT WORK WITH ORACLE 7.3 (NOR VICE VERSA)

Due to the many changes introduced in Oracle 7.3, the compatibility was broken with applications that were compiled with Oracle 7.1 or 7.2. Thus, for CMVC 2.3.1 we provide whenever possible two versions for Oracle:

- One natively compiled with 7.1 and which also works with 7.2. This version does not work with 7.3.

- Another natively compiled with 7.3 and which does not work with 7.1 or 7.2.

Note: Because Oracle 7.1, 7.2 are not going to be maintained by Oracle after 31-Dec-1998, we may provide only the CMVC server for Oracle 7.3.

MIGRATION FROM CMVC 2.2 USING ORACLE 6 TO CMVC 2.3 USING DB2

This section describes the overall migration scenario from CMVC 2.2 using Oracle 6 to CMVC 2.3.0 using DB2 V2 on AIX 3.2.5. For mode details, see the appendices B and D from the CMVC 2.3 Server manual.

This scenario assumes that the host is the same, and that DB2 V2 is installed in the host.

1. Migrate from CMVC 2.2 for Oracle 6 to CMVC 2.2 for DB2 V2 on AIX 3.2.5.

   This is necessary because there is no support for Oracle 6 on CMVC 2.3, because Oracle dropped support for version 6 in December 1994. CMVC 2.3 supports only Oracle 7.

2. It is necessary to perform db2bind after the migration to DB2.
3. Verify that the migration is successful before continuing.
4. Upgrade from CMVC 2.2 for DB2 V2 to the latest version of CMVC 2.3 for DB2.
5. It is necessary to perform db2bind in the new family.
6. Verify that the upgrade was successful.

MIGRATING FROM ORACLE 6 IN AIX 3.2.5 TO ORACLE 7.1 IN AIX 4.1

The migration route that we recommend is as follows:

1. Migrate from CMVC 2.1/2.2 using Oracle 6 on AIX 3.2.5 to CMVC 2.1/2.2 using Oracle 7.0 on AIX 3.2.5.

   Notes:

   a. Use the Appendix C of the CMVC 2.3 Server manual.
   b. There is no support in CMVC 2.3 for Oracle 6.

2. Migrate from CMVC 2.1/2.2 Oracle 7.0 on AIX 3.2.5 to CMVC 2.3 Oracle 7.0 on AIX 3.2.5.

   This is very easy. See Appendix B of the CMVC 2.3 Server manual.

3. Migrate from CMVC 2.3 Oracle 7.0 on AIX 3.2.5 to the latest version of CMVC 2.3 (2.3.0.25) Oracle 7.1 on AIX 3.2.5.

4. Migrate from CMVC 2.3.0.25 Oracle 7.1 on AIX 3.2.5 to CMVC 2.3.1.4 Oracle 7.1 on AIX 4.1.

PREPARING TO MIGRATE TO VISUALAGE TEAMCONNECTION VERSION 3

This chapter provides a brief description of VisualAge TeamConnection Enterprise Server Version 3, the successor of CMVC. It describes some recommendations for the migration of a CMVC family into VisualAge TeamConnection.

In some cases it is necessary to clone the CMVC family from one host to another in order to work with CMVC 2.3.1, which is necessary to migrate to VisualAge TeamConnection. For more details see “Cloning/migration of a CMVC family from one host to another” on page 25 and “Migration tips when moving from one version of CMVC to another” on page 31.

WHY VISUALAGE TEAMCONNECTION?

CMVC is an industrial strength source code library with extensive support for tracking of changes through defects and features, and a configurable and extendible process model (by using user exits). However, the role of the development library is evolving.

VisualAge TeamConnection Enterprise Server Version 3 is IBM's next generation of library management. Actually, TeamConnection is much more than a library. TeamConnection extends the CMVC process model to include a tightly integrated build facility, a highly customizable packaging facility, and support for network distribution of your packaged code.

Furthermore, TeamConnection incorporates an object repository and API's to allow tools to store their data (that is, large grained objects such as files and small grained objects such as data field information created by 4GL languages like IBM's VisualAge Generator in TeamConnection.

These integrated tools can take advantage of TeamConnection's releases, defects, features, etc. so that all of the developer's data is managed and versioned together. This dramatically reduces the overhead of using a collection of tools in the development process.

VisualAge TeamConnection provides a Web client that allows the users to interact with a Web browser to issue TeamConnection commands. Also, TeamConnection offers a Lotus Notes federation in which a Lotus Notes database can be used to track things such as design documents and test cases, and then track TeamConnection features and defects.

Finally, the object repository interface allows for tools used by anyone in your entire development team (document writers, marketing specialists, testers, etc.) to manage and version their data in one place.

Now everyone's data can evolve without lots of manual translation. For example, a software architect's specification can be tracked as the software developer writes code to match the specifications. This transition is trackable and auditable.

CMVC and it successor product VisualAge TeamConnection Enterprise Server Version 3 have a lot in common but they are not compatible between each other. That is, you cannot use a CMVC client with a TeamConnection server, nor a TeamConnection client with a CMVC server.

This means that if you want to use TeamConnection, then you have to migrate your CMVC family into TeamConnection.

MIGRATION FROM CMVC TO VISUALAGE TEAMCONNECTION

The TeamConnection migration utility "migcmvc" uses CMVC client commands. The only requirement from TeamConnection is that the CMVC client be installed on the same system as the TeamConnection server for a migration to occur.

In principle, any CMVC client, version 2.2 or later, will be able to support TeamConnection migration. However, you must upgrade your version of the CMVC family server to the latest supported version of CMVC 2.3.1 before migrating to TeamConnection.

The following shell script samples will help you to prepare the CMVC family for migration:

- “Redefine ChangeView to facilitate the migration to TeamConnection” on page 85
- “Complete a release prior to migration” on page 85
- “Reassign user's work and delete users” on page 85
- “Delete a release from a family” on page 86
- “Converting SCCS keywords to TeamConnection Keywords” on page 86

It is recommended that you perform the migration from CMVC to VisualAge TeamConnection by following the instructions from the technical report 29.3113, "Migration from CMVC 2.3.1 to VisualAge TeamConnection Enterprise Server Version 3". It is available from:

`ftp://ftp.software.ibm.com/ps/products/teamconnection/papers/trcm2tc3.pdf`

DEALING WITH COMMON MIGRATION PROBLEMS

This chapter provides some guidance on how to deal with common migration problems.

HOW TO DEAL WITH MESSAGES 0011-110 AND 0011-111

Question:

When I try to start the CMVC family for the first time after the migration procedure has been completed, I am getting one of the following error messages saying that one element of the Sequence table is smaller than some of the values found in another table:

```
0011-110 The value of lastSerial where name='source' ...

0011-111 The value of lastSerial where name='general' ...
```

Answer:

The Sequence table is extremely important to CMVC because it provides the next number/id to be used for the internal ids for the objects. One main cause of this corruption of the Sequence table or related tables is when the user directly alters the database bypassing CMVC; another possible cause is an error during the migration.

During the migration steps it might be possible that the original Sequence table of the target CMVC family database has the default values (defect=0, source=0, general=2) when a new family is created. If this is the case, then when loading the tables from the source database, some DBMS will not insert the appropriate values because of the duplicate key constraint, and thus the correct values are not loaded.

To solve this problem, restart the migration steps and delete the values for the Sequence table. In that way, during the import/load/restore step the correct values will be inserted. Otherwise, you may want to manually update the Sequence table in the target database with the correct values from the Sequence table from the source database.

If you get this error message meanwhile you are not doing any migration steps, then our recommendation at this stage is to take a backup now of both the database and of the file system of the family. Then, use the latest good backup and reestablish your family database and file system. If at this stage the server is OK, then keep using it.

HOW TO CREATE A HOST LIST ENTRY DIRECTLY INTO THE DATABASE

Question:

A common mistake when migrating CMVC families from one host to another is to fail to add before the migration a host list entry for the new target host. The symptom is the error message 0010-057 when trying to run a CMVC command in the newly migrated CMVC family.

At this point, the CMVC family administrator "has painted himself into a corner", because in CMVC it is required to have a valid host list entry in order to do invoke CMVC commands.

Answer:

To overcome this problem it is necessary to add a host list for the CMVC superuser (which by default has the internal userid of 1) by using SQL statements directly to the database, bypassing CMVC. By the way, this is the method used by the 'mkdb' when creating the CMVC database.

The following sequence is for DB2, but it is similar for other DBMSs:

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

   ```
   $  db2 connect to $CMVC_FAMILY
   ```

3. Use interactive SQL. In DB2 (command line processor) type:

   ```
   $  db2
   ```

4. Issue the following insert command (one single line):

   ```
   db2 => insert into Hosts (userId, name, login, address)   \
          values (1,'$CLIENT_HOSTNAME', '$CLIENT_LOGIN',0)
   ```

   The "userId=1" represents the first CMVC userId, which is the superuser when the CMVC family was created.

   The variables CLIENT_HOSTNAME and CLIENT_LOGIN represent the hostname and the system login from where the superuser will login.

   The CLIENT_HOSTNAME should be the string that is the complete name returned by the command "host". For example, if your host is "carcps06", then do "host carcps06" and you will get something similar to:

   ```
   carcps06.raleigh.ibm.com is 9.67.238.18
   ```

   Then, the CLIENT_HOSTNAME should be "carcps06.raleigh.ibm.com".

   You can also issue the "ping" command, which also shows the complete host name with the domain name.

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

   ```
   db2 => commit work
   ```

6. Query the Hosts table to ensure that the data is correct:

   ```
   select userid,login,name from Hosts where userId=1
   ```

7. Terminate the interactive SQL session:

   ```
   db2 => quit
   ```

8. In case of DB2, disconnect:

   ```
   $  db2 terminate
   ```

AFTER MIGRATION I CANNOT DO LEVEL -COMPLETE OR DEFECT -ASSIGN

Question:

After migrating a CMVC family, one or more of the following DB2 errors are shown when trying to perform Level -complete or Defect -assign:

```
SQL0051N
SQL0305N
SQL0503N
SQL1328N
```

Answer:

The most likely cause is that the record with id=0 in the Releases table does not have the proper values. To fix this problem do the following:

1. Invoke the DB2 Command Line Processor (CLP) and specify that the semicolon (;) is the termination character:

   ```
   $ db2 -t
   ```

2. Verify that there is a record with id=0 in the Releases table:

   ```
   db2 => select id, compId, userId, relSize from Releases where id=0;
   ```

   The output should look like this:

   ```
   ID       COMPID   USERID      RELSIZE
   -------  -------  --------  ---------
         0        0         0          0

     1 record(s) selected.
   ```

3. If the data for id=0 does not have "0" in the above fields, then do:

   ```
   db2 => update Releases set compId=0, userId=0, relSize=0 where id=0;
   ```

4. If there is no data for id=0, then insert a new record:

```
db2 => insert into Releases (id, name, compiId, binding, description,
db2 (cont.) => userId, addDate, dropDate, lastUpdate, relSize)
db2 (cont.) => values (0, null, 0, null, null, 0, null, null, null, 0)
db2 (cont.) => ;
db2 => commit;
```

5. Exit from the DB2 CLP and terminate the connection.

```
db2 => quit;
$ db2 terminate
```

AFTER MIGRATION I DO NOT SEE THE HISTORY WHEN DOING FILE -VIEW -LONG

Question:

After migrating a CMVC family, I do not see the version history when performing File -view -long.

Answer:

The most likely cause is that the record with id=0 in the Versions table does not exist. To fix this problem do the following:

1. Invoke the DB2 Command Line Processor (CLP) and specify that the semicolon (;) is the termination character:

   ```
   $ db2 -t
   ```

2. Verify that there is a record with id=0 in the Versions table:

   ```
   db2 => select id, previousId, sourceId, userId from Versions where id=0;
   ```

   The output should look like this:

   ```
   ID          PREVIOUSID  SOURCEID    USERID
   ----------- ----------- ----------- -----------
             0           0           0           0

     1 record(s) selected.
   ```

3. If there is no data for id=0, then insert a new record:

   ```
   db2 => insert into Versions (id, previousId, sourceId, userId, rawLines,
   db2 (cont.) => codeLines, commentLines, remarks, changeDate, SID, verSize)
   db2 (cont.) => values (0, 0, 0, 0, 0, 0, 0, null, null, null, 0)
   db2 (cont.) => ;
   db2 => commit;
   ```

4. Exit from the DB2 CLP and terminate the connection.

```
db2 => quit;
$ db2 terminate
```

AFTER MIGRATION I CANNOT LIST FILES WITH UNCOMMITTED CHANGES

Question:

After migrating a CMVC family, I cannot list the files with uncommitted changes; they are not included in the output of the Report command.

Answer:

The most likely cause is that the record with id=0 in the Levels table does not exist. To fix this problem do the following:

1. Invoke the DB2 Command Line Processor (CLP) and specify that the semicolon (;) is the termination character:

   ```
   $ db2 -t
   ```

2. Verify that there is a record with id=0 in the Levels table:

   ```
   db2 => select id, releaseId, userId from Levels where id=0;
   ```

   The output should look like this:

   ```
   ID          RELEASEID   USERID
   ----------- ----------- -----------
             0           -           -

     1 record(s) selected.
   ```

3. If there is no data for id=0, then insert a new record:

   ```
   db2 => insert into Levels (id, releaseId, userId, addDate, commitDate,
   db2 (cont.) => name, lastUpdate, state, type)
   db2 (cont.) => values (0, null, null, null, null, null, null, null, null)
   db2 (cont.) => ;
   db2 => commit;
   ```

4. Exit from the DB2 CLP and terminate the connection.

   ```
   db2 => quit;
   $ db2 terminate
   ```

AFTER UPGRADE TO 2.3.1, THE DATE FIELDS ARE TRUNCATED

Question:

I just upgraded to CMVC 2.3.1 and I performed the dbSetDate tool to convert the dates from from yy/mm/dd to yyyy/mm/dd.

During my testing, I uncovered the following irregularity in the section 'versions' in the output of File -> Show Details (File -long -view):

```
versions:
     user     date     SID
     -------- -------- ----------------------------------------
     mannk    1998/12/ 1.12.1.2
```

It appears the date field in the output is fixed to a length of 8 and is no longer big enough to support the new size of the date.

Is this OK? If not, how can I fix it?

Answer:

The problem is that you are using the new 2.3.1 code for CMVC, but the message catalog (cmvc.cat) for 2.3.0 is found first in the NLS path and thus, the date fields are truncated.

Ensure that you have installed the message catalog for CMVC 2.3.1 for the server or the client in your machine. Use the following commands to find out what is the version of the message catalogs:

- For the CMVC server:

  ```
  Report -testServer
  ```

- For the CMVC client:

  ```
  Report -testClient
  ```

The result should be something like this:

```
The message catalog is available (Version 2.3.1.4, SID=1.285.2.175).
```

If the Version says 2.3.0.x then you need to adjust the NLSPATH to reflect the location of the 2.3.1 message catalog file.

0010-256 ERROR MESSAGES AFTER MIGRATING FROM CMVC V1.1 TO V2.X

There is a potential problem after migrating a CMVC family from V1.x to V2.x in which the command "Defect -configInfo" will not work (error message 0010-256 for txDefectConfig in the client side, and error 0010-636 with a database error indicating more than 1 row).

It might be possible in CMVC 1.x to have multiple defaults in one configuration item type (from config.ld) which were loaded into the Config table and this is the root problem, because in CMVC 2.2, if there is a default, it should be only 1 and no more.

In CMVC 1.1, there was no checking for this situation, and the migration step does not touch this table and does not reload it fresh from the source config.ld file.

Fixing CMVC V1.1 to V2.x migration problems with "chcfg"

Immediately after the migration is over, but before starting the CMVC daemons, it is necessary to update the file config.ld to avoid multiple defaults for an item, and then to use "chcfg" to reload the items for the Config table (which must be successful).

RENAMING AND MOVING A CMVC FAMILY

RENAMING A CMVC FAMILY

Scenario:

1. Current CMVC family is on host using a database supported by CMVC (in this example we will use Oracle 7).
2. We want to keep the CMVC family in the same host, but we want to change the family name.

Recommendation:

Changing the name of a CMVC family is rather straight forward, but there are a couple of things to watch out for, so here is our recommended procedure.

By the way, this procedure is particularly useful when you wish to move data from the system (default) database to separate database files.

Notes:

1. The following procedure can also be used for migration of CMVC families when it is desired to change operating systems, versions of CMVC, and/or versions of a database in one step. However, it takes longer than following the procedures documented in the CMVC Server manual.
2. For details on how to use DB2 export and DB2 import to move a database from one place to another, see the TR 29.3088 "Moving a VisualAge TeamConnection Version 3 Family". The main concepts discussed in the TR also apply to a CMVC family. See “Other related technical reports” on page 7.

The recommended sequence is shown below:

1. Export the database tables and indexes of the current database. A sample syntax is provided in the CMVC Server manual in page 195, "Exporting from Oracle 6" in Appendix C.
2. Drop the current copy of the database.

   This is a simple alternative to actually renaming the tables, indexes, etc in the existing instance.

   a. Execute: rmdb

b. If using Oracle, drop the table space and the index space pointed to by environment variables: ORACLE_TBLSP and ORACLE_NDXSP.

3. Change the name of the family in /etc/passwd.

   You can use the appropriate system administration tool for the appropriate platform. For example, in AIX you can use smit.

4. Change the ownership of all files in the family account.

   a. Login as root.
   b. Issue the following commands:

   ```
   $ mv /home/FAMILY_NAME /home/NEW_FAM_NAME
   $ find /home/NEW_FAM_NAME -exec chown NEW_FAM_NAME {} \;
   ```

   Where FAMILY_NAME is the old family name and NEW_FAM_NAME is the new family name.

   c. Log out as root.

5. Update the .profile of the family user ID.

   Update the environment variables that reference the family name, such as CMVC_FAMILY, CMVC_TOP (if used), and PATH.

   Also, this would be a good time to compare your .profile to our sample profile in /usr/lpp/cmvc/install/profile.<db>, where <db> is a database name.

6. Find all SCCS archive "p." files in the vc tree and change the family name in these files:

   a. Log into the family user ID.
   b. Issue the following command:

   ```
   $ find /home/NEW_FAM_NAME/vc -type -name "p.*" -print e
     while read file
     do
        awk 'BEGIN{print "%s/'${OLD_FAMILY}'/'${FAMILY}'/";    \
        print "wq"}' /dev/null e ex ${file} > /dev/null 2>&1
    done
   ```

   c. If you would prefer to save the old copies of each file, you can use the following command:

   ```
   $ for i in  find /home/NEW_FAM_NAME/vc -type -name "p.*" -print
     do
        mv $i ${i}.old
        sed "s/${OLD_FAMILY}/${NEW_FAMILY}/g" ${i}.old > $i
        # rm ${i}.old
     done
   ```

7. Update /etc/hosts and /etc/services with the new family name.

8. Create the new database using Oracle:

a. Create a new table space and a new index space.
b. Update the .profile to use the new index and table space.
c. Issue: mkdb

Note: If you originally issued `mkdb -d` to create the database you will either need to do it here, or perform the step 8e below.

d. Delete the preloaded table entries created by mkdb:

```
$ sqlplus family/passwd
> delete from users;
> delete from hosts;
> delete from config;
> delete from interest;
> delete from authority;
> delete from sequence;
> delete from components;
> delete from compmembers;
> delete from levels;
> delete from releases;
> delete from versions;
> delete from cfgrelproc;
> delete from cfgcomproc;
> commit;
> quit;
```

e. Run chfield for customized configurable fields.
   - chfield -object Defect -source $HOME
   - chfield -object Feature -source $HOME
   - chfield -object File -source $HOME
   - chfield -object User -source $HOME

f. Temporarily drop the indexes to speed up the import of data. The following shell script can be used to create the shell script shown in Figure 1 on page 74:

```
#!/usr/bin/ksh
OUTPUT=drop.index.sh

# Create a file with instructions for converting the CMVC script
# used to create indexes into one that drops them.
# It has Oracle specific syntax.
cat <<EOF!  > sed.list
s/unique //
s/create/drop/
s/$/;/
EOF!

# The file index.db is specific to Oracle.  There are others for
# the other databases supported by CMVC.

grep create $CMVC_HOME/install/index.db e sed -f sed.list > $OUTPUT
echo "commit;
quit;" >> $OUTPUT

# Run SQL script to drop indexes
sqlplus $ORACLE_DBA @${OUTPUT}
rm $OUTPUT sed.list

exit 0
```

Figure 1. Script to create and run SQL instructions to drop indexes

g. Import data.

   Please consult your database documentation for the procedure to import data.

   CMVC provides the command to use with Oracle 7 in page 196, step 5 of "Importing to Oracle 7" in Appendix C of the CMVC Server manual.

   If problems occur on a particular table, consult the database documentation for the procedure to re-try just that table.

   If problems occur importing the Defects, Features, Files or Users tables, see the step 8e on page 73.

   If all else fails, drop the table and then import it again (the table will be imported into the default table space).

h. Recreating indexes.

   You need to run a slightly modified version of the file that defines the indexes in /usr/lpp/cmvc/install, such as index.db.

   Please consult your database documentation for procedures. CMVC provides a script that modifies the index.db to use with Oracle 7 in step 6 of "Importing to Oracle 7" in Appendix C of the CMVC Server manual.

MOVING A CMVC DATABASE FROM A DB2 INSTANCE TO A NEW ONE

Scenario:

1. The database for a CMVC family is stored in one DB2 instance.
2. We want to move the database for the CMVC family from the current DB2 instance to a new DB2 instance.

Recommendation:

The recommended sequence to move an existing family from one DB2 instance to another is as follows:

1. Create and start the new DB2 instance.
2. Login as the family account.
3. Stop the family and make a backup of the database. For details see “How to make a complete backup of your CMVC family” on page 9.
4. Change the family's DB2INSTANCE environment variable to point to the new instance, and change DB2_DBPATH to point to the new location of the family in the new instance (if needed).

   Logoff and login again to refresh the environment variables.
5. Make the family user ID a member of the new instance's primary group in order to have SYSADM authority on the database.
6. Restore the database. For details see “How to restore a CMVC family from a complete backup” on page 14.
7. If you do a test run first, you will have to uncatalog the new database before you can restore it again.
8. Restart the family.

This sequence will leave all the old database files on the system and leave the old database cataloged on the old instance. At some stage you will want to delete all these old database files and to uncatalog the old database from the original instance.

MIGRATING DATA FROM LIBRARY SYSTEMS INTO CMVC

CMVC uses the Unix SCCS utility as the versioning mechanism for the files. However, the end users of CMVC do not deal directly with SCCS.

Some customers already use SCCS in a direct way to store versions of files, and when these customers want to migrate their SCCS versioning files into CMVC, they need to perform a migration process; they cannot just simply copy their existing SCCS files into the CMVC versioning tree ($HOME/vc).

BRIEF EXPLANATION OF SCCS

The SCCS library system is provided free with most Unix operating systems. It does not have tracking, it cannot handle binary files and it does not provide a GUI; therefore this tool is easy to outgrow. That is why CMVC provides a migration from SCCS.

Of course, there are many other simple library systems whose users would benefit from CMVC.

For a useful shell script to import existing SCCS files into CMVC, see “Import a release from directory structure” on page 86

Explanation of migration from SCCS to CMVC

Note: The following explanation was given by Gary Warner, Software Tools Support, IBM Austin.

All of the migration scripts for SCCS files (Filemap, Fileimport, Filemigrate, and xecit) are found in the /usr/lpp/cmvc/bin directory on the CMVC client.

You can use these SCCS files provided you import or migrate them into the CMVC development environment. The SCCS File Import and Migration Tools provide a means by which CMVC users can bring SCCS text files into the CMVC development environment.

To use these tools, the destination CMVC family must use SCCS as its underlying version control mechanism.

The SCCS File Import and Migration tools are documented in the 'Bringing SCCS Files Under CMVC Control' section of the IBM CMVC Server manual. Refer to the discussion in this section for the advantages/disadvantages of each method.

When SCCS files are imported, their version number in the CMVC development environment becomes 1.1. We recommend that users keep a copy of their map files so that they can look back to which version of each file they imported.

When SCCS files are migrated, all of the versions of the file are brought into the CMVC development environment. The user specifies which version is to be known as the current version in a release. The user does not have to assign each branch of an SCCS file to a release; they can do this later.

When SCCS files are migrated into the CMVC development environment, the version history and remarks will be captured. If a CMVC user ID exists for the user who created the SCCS version, then this information will be retained. If a CMVC user ID does not exist for the user who created the SCCS version, then the information will be recorded as the user who is performing the migration.

Common files are supported with the import and migration tools if all releases that are to contain common files are processed by the Fileimport or Filemigrate scripts simultaneously. This is accomplished by supplying more than one map file name as parameters to these scripts.

The Migrate command is used to bring an SCCS file into CMVC. This is essentially File -create with the command and action changed to Migrate -migrate and with an additional -version flag.

The File -create command creates an s. file at version 1.1 from the input file. The Migrate command creates all versions in an input s. file and makes one of them the "current" version.

The Migrate command uses a shell script named sclean which is in the CMVC bin directory (with cmvcd) to "clean" the file and to tell the Migrate command what versions are in it. The "cleaning" task means to remove any flags that restrict access of the file to certain logins, and sets a flag to allow multiple gets for edit.

The sclean script writes the version information to stdout which is then read by the Migrate command. This information is:

```
#versions
deltanum previous SID date time username
remark lines, if any
<a blank line>
repeat for other versions in deltanum order.
```

The sclean script gets this stuff from the header of the SCCS file. The SCCS utility numbers each delta as it is created. This is "deltanum". The "previous" is deltanum of the version from which the given version was created, with previous = 0 for the first delta. The "username" is

the login which created the delta. If this matches a CMVC login, Migrate uses that login as the creator. If it does not, the user issuing the Migrate command gets the credit.

WHAT PROBLEMS CAN BE ENCOUNTERED WITH THE IMPORT AND MIGRATION TOOLS?

Question:

What problems can be encountered with the Import and Migration tools?

Answer:

Some common situations that cause problems with these migration tools are shown below:

- The file.import or file.migrate files do not have execute permission.
- The Filemap, Fileimport or Filemigrate scripts are being run by a user who has insufficient AIX user authority to create files within the current directory or read the SCCS files in the directories in which they reside.
- The scripts are being run from a client that does not have the CMVC client.
- The components and releases have not been created in the CMVC environment yet.
- The files being imported or migrated already exist in the CMVC environment.
- The user running the import or migration scripts has insufficient CMVC access authority (FileAdd and FileLink are required).
- A defect or feature is in the wrong state.
- A track or a fix record is in the wrong state.
- The CMVC server's vc tree is full.

CHANGES TO THE ERROR HANDLING OF FILE.MIGRATE AND FILE.IMPORT

CMVC 2.2 and initial versions of CMVC 2.3 made use of a utility for performing error checking on the file.migrate and file.import script files. However, most users forgot to use xecit and were unable to determine whether or not their data was properly loaded into the CMVC family. As a result, later versions of CMVC 2.3 now puts the error checking directly into the file.migrate and file.import script files.

The current file.migrate and file.import script files, generated from FileMigrate and FileImport, include error handling. If an error occurs, then a file.migrate.err or file.import.err file will be generated and will identify the errors. Except in extreme cases, this file will be very small, if it is created at all.

If a file.migrate.err or file.import.err file is generated, you can use the "xecit" command to execute each line in these files and to capture the errors in an executable file like file.migrate.err or file.import.err. The utility xecit is documented on pages 131 and 132 in the CMVC Server manual.

You may find xecit useful for any script where you do not want to include the error checking code in it.

HOW SCCS ARCHIVES RELATE TO CMVC RELEASES

It has been asked why a single SCCS archive cannot always be migrated into CMVC in one release. The basic answer is that CMVC can only migrate one branch of the version tree in an SCCS archive in a single release.

For example, if s.foo archive for the foo file has file versions 1.1, 1.2, 1.3 and 1.2.1.1, then a conscious decision was made that 2 different file versions would be created as successors to version 1.2. A CMVC release can contain version 1.1, 1.2 and 1.3, but would not be able to contain 1.2.1.1 because it is conflicting with version 1.3. Therefore, a second release would need to be migrated that would contain versions 1.1, 1.2 and 1.2.1.1. In both cases, versions 1.1 and 1.2 could be linked between both releases.

In either case, the file foo would be common to both releases because CMVC would internally use one archive to store the versions.

MIGRATING FROM ANOTHER LIBRARY TO SCCS

The easiest way to get from these libraries to CMVC is by migrating first to SCCS, then from SCCS to CMVC.

Migrating from RCS to SCCS

For example, it has been reported to us that there is an RCS to SCCS conversion tool available at: ftp://www.cyclic.com/pub/cvs/contrib/cvs-1.8.5 as part of the package cvs-1.8.5.tar.gz. If you use gnu zip to unzip to file, you will find a Korn shell script for converting from RCS to SCCS in the contrib directory. You can get to this site by using http://www.cyclic.com.

Bringing PVCS files under CMVC control

Question:

Is there a recommended way to bring PVCS files (such as from OS/2) under the control of CMVC using SCCS (such as on AIX)?

Answer:

In order to provide some advice for PVCS migration, it is important to understand the migration from SCCS:

There are several scripts which are used to help build the proper arguments for the CMVC Migrate commands to migrate files from an SCCS source tree.

The CMVC Migrate command is used to bring an SCCS file into CMVC. This is essentially "File -create" with the command and the action changed to "Migrate -migrate" and with an additional "-version" flag.

The CMVC "File -create" command creates an "s." file at version "1.1" from the input file. The CMVC "Migrate -migrate" command creates all versions in an input "s." file and makes one of them the "current" version.

The CMVC Migrate command uses a CMVC shell script named "sclean" to "clean" the file and tell Migrate what versions are in it. The term "cleaning" means to remove any flags restricting access of the file to certain logins and to set a flag to allow multiple gets for edit.

The script sclean writes the following version information to stdout which is then read by the Migrate command:

```
#versions
deltanum previous SID date time username
remark lines, if any
<a blank line>
repeat for other versions in deltanum order.
```

The script sclean gets this stuff from the header of the SCCS file. SCCS numbers each delta as it is created and this is "deltanum". The field "previous" is deltanum of the version from which the given version was created, with previous = 0 for the first delta. The field "username" is the login which created the delta. If this matches a cmvc login, then Migrate uses that login as the creator. If it does not, then the user that is issuing the Migrate command gets the credit.

So, if you can make a PVCS version of sclean and put it in the same directory as sclean, then you should be able to use the Migrate command to bring PVCS files into CMVC.

APPENDIX A. OBTAINING THE MIGRATION SHELL SCRIPTS AND LATEST CMVC CODE

The shell scripts described in this technical report as well as the latest CMVC code 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 the section "Migration tools" or the appropriate operating system.

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:
4. cd cmvc/doc/tr
5. ascii
6. get <filename>
7. 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/tr

4. ascii
5. get <filename>
6. quit

From the directory ps/products/cmvc you may choose the subdirectory for the desired operating system, in order to access the latest CMVC code.

APPENDIX B. MIGRATION SHELL SCRIPTS

SHELL SCRIPTS TO PREPARE THE MIGRATION FROM CMVC TO TEAMCONNECTION

Redefine ChangeView to facilitate the migration to TeamConnection

The name of the shell script is:

`xxxUpdateChangeView.ksh`

Where xxx is the name of the DBMS: db2, oracle, sybase or informix.

This utility will redefine ChangeView in order to migrate the change history to TeamConnection.

Complete a release prior to migration

The name of the shell script is:

`release.complete.ksh`

This utility will complete work in a given release so that all file changes are committed.

Reassign user's work and delete users

The name of the shell script is:

`reassign.work.ksh`

This utility will reassign incomplete work from one active CMVC login to another active CMVC login. Incomplete work can be defined as:

```
defects owned/originated
features owned/originated
verification
components
releases
levels
environments
test records
size records
approval
approver
tracks
fix records
notifications deleted
access deleted
login deleted
```

Delete a release from a family

The name of the shell script is:

`release.delete.ksh`

This utility will delete work in a given release. It will also identify a list of vc (SCCS/PVCS) files that are not related to any other release that can be removed to save disk space.

Converting SCCS keywords to TeamConnection Keywords

The name of the shell script is:

`keywordconversion.migcmvc`

This sample script converts SCCS keywords to TeamConnection keywords for text files that exist for a given RELEASE (CMVC_RELEASE) under a specified directory (CMVC_TOP).

The script currently supports file extensions .c and .ksh. To add additional extensions, you should modify the function check_extension to add the appropriate case statement.

MISCELLANEOUS SHELL SCRIPTS

Import a release from directory structure

The name of the shell script is:

`file.import.ksh`

This utility will load files into CMVC from a given root directory. It requires that the user should provide a mapfile that maps the pathnames of the files to the components where the files are to be loaded into.

APPENDIX C. 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
HP, HP-UX	Hewlett-Packard Company
IBM, OS/2, AIX, VisualAge Generator, CMVC, DB2, Universal Database, VisualAge, TeamConnection	IBM Corporation
INFORMIX	Informix Inc.
OSF, OSF Motif	Open Software Foundation, Inc.
ORACLE	Oracle Corp.
Sun, SunOS, OpenWindows, Solaris	Sun Microsystems Inc.
SYBASE	Sybase Inc.
Unix, USL	Unix System Laboratories, Inc.

END OF DOCUMENT °