

**BSTAR 'Backup and Recovery for Library Units' (BRLU)
Version 6s**

Title: *BSTAR 'Backup and Recovery for Library Units' (BRLU)*

File: *BSTAR BRLU Product Manual.pdf*

Document Date: *15/10/2010 13:39 Document Version: 6.7.0.0*

Table of Contents

1.	BSTAR product suite.....	4
2.	BRLU and library units.....	5
3.	BSTAR packaging and licensing.....	6
4.	Simplified BSTAR installation	8
5.	Thread safety and multithreaded job support	9
6.	BSTAR runtime – Option *BASE.....	10
6.1	System requirements.....	10
6.2	Installation.....	10
6.3	Security settings	10
6.4	Performance	11
6.5	User Guide	11
6.5.1	Common runtime environment (BSTAR mode)	11
6.5.2	Object “protection” components	13
6.5.3	Journal checkpoints	15
6.5.4	Commands and menus	15
6.5.5	Cleanup	16
6.5.6	Exits defined.....	17
6.5.7	“Submit concurrent batch jobs” framework.....	17
7.	Asynchronous Backup and Restore – Option 1.....	19
7.1	System requirements.....	20
7.2	Installation.....	20
7.3	Security settings	20
7.4	User Guide	20
7.4.1	Commands and menus	21
7.4.2	Background backup server.....	23
7.4.3	Exits defined.....	23
8.	BSTAR Command Transformation – Option 2.....	24
8.1	System requirements.....	24
8.2	Installation.....	24
8.3	Security and audit settings	25
8.4	User Guide	25
8.4.1	Example	26
8.4.2	Parsed substitution variables	27
8.4.3	Odd transformation.....	28
8.4.4	Commands and menus	29
8.4.5	Compatibility.....	30
8.4.6	Exits defined.....	30
8.4.7	Service trace.....	31
9.	Recovery for Library Units – Option 3.....	32
9.1	System requirements.....	32
9.2	Installation.....	32
9.3	Security settings	33
9.4	User Guide	33
9.4.1	Recovery in High Availability Environments	33
9.4.2	Menu	35
9.4.3	Commands	36
10.	Help panels.....	38

Trademarks

Any trademarks and product or brand names referenced in this document are the property of their respective owners.

1. BSTAR product suite

BSTAR is a suite of licensed program offerings for IBM i(i5/OS) operating system designed to improve performance characteristics of long running batch processes. Performance improvement is generally achieved by implementation of multiple parallel programming technologies.

All BSTAR tools use the same runtime foundation functions. BSTAR suite contains the following products packaged as options of 4F44RCS licensed program:

- BSTAR runtime functions (option *BASE)
- BSTAR asynchronous backup and restore (option 1)
- BSTAR system command substitution (option 2)
- BSTAR recovery for library units (option 3)
- BSTAR generic multi-streaming toolkit (option 4)

Options 1, 2 and 3 have traditionally been packaged as BRLU (Backup and Recovery for Library Units) utility. This manual only describes BRLU functions. There is a separate user manual for BSTAR generic multi-streaming toolkit.

2. BRLU and library units.

BRLU is a group of tools for IBM System i designed to improve recoverability characteristics of long running batch applications while at the same time boosting their performance.

A library unit is a group of IBM i(i5/OS) libraries having common backup and recovery policies. In other words, libraries from this group are usually backed up and recovered as a group.

A library unit may be thought of as a sub-database, a self-contained subset of IBM i(i5/OS) objects representing the data store of a certain application.

BRLU is a tool that can be used to define very efficient recovery policies for library units. It can also greatly improve efficiency of the existing policies.

BRLU implementation, as a rule, does not involve any programming effort.

The BRLU approach to recovery is to rely not so much on regular backups but on IBM journal checkpoints and the capability, provided by IBM i(i5/OS) system software, of rolling back database and data area changes using IBM journal data. BRLU provides a simple interface for creating IBM journal entries that can be used as recovery checkpoints, and another interface – for rolling back all protected objects to any of the previously taken checkpoints.

The journal checkpoint recovery solution implemented in BRLU makes it possible to submit usually synchronous tape backup operations for asynchronous execution. Thus, a lot of precious batch time is saved. The mode of existing backups can be changed automatically, without any of the existing programs being modified; IBM save-while-active facility used by BRLU guarantees that asynchronous backups generate tapes (or disk files) absolutely identical to those created in synchronous mode.

Another BRLU strength is the ability to recover High Availability configurations of library units after logical errors in long running batch processes. Without BRLU such errors usually require full recovery from backups for both the production and the hot standby systems, which can be problematic, especially in case of the hot standby system being geographically remote.

Last but not least, BRLU can help hook existing applications using traditional IBM i(i5/OS) backup facilities to 3rd party (e.g. BRMS) archival and backup management software.

BRLU is at its best when it is desirable to implement a new recovery policy for an existing application without changing any of the application components.

3. BSTAR packaging and licensing

BSTAR is packaged as five options of 4F44RCS licensed program.

*BASE option provides framework functionality for other four options.

Option 1 contains “asynchronous backup” functionality. It allows users running under protection of the *BASE option to automatically have their backup commands intercepted and executed asynchronously.

Option 2 contains general “command substitution” functionality. Using Option 2 it is possible to define different IBM i(i5/OS) backup and recovery command substitution rules. These rules are defined on a per unit basis and applied at runtime.

Option 3 contains recovery functionality. Using Option 3 it is possible to program complex recovery scenarios including multiple BSTAR commands by example and then, when required, invoke those scenarios by simply choosing an option from a menu. Option 3 can be also used to improve performance of the recovery process.

Option 4 contains generic multi-streaming toolkit. Using Option 4 it is possible to split-stream (or multi-stream) any process executing in the ILE environment, the main purpose of split-streaming being improvement of process runtime.

Starting from V6R3M0 all BSTAR options are no longer licensed separately, but separate license installation programs are still to be obtained for each IBM i(i5/OS) logical partition the option will be used in. A license request must include the serial number of the IBM POWER server.

The names of the save files and programs are SYSTEM1-SYSTEM4, depending on the option. License keys are not generated or distributed for option *BASE.

Each of options comes with a trial license implemented as a grace period. Trial licenses do not restrict product functionality. The following grace periods apply:

- Option *BASE is distributed as free software
- 90 days for Options 1,2 and 3
- 30 days for Option 4. Grace period for this option is activated only when the number of streams defined is greater than 2.

An additional post-grace period of 21 days is defined for options 1-4 of the product. When a temporary license expires product functionality is disabled, e.g. multi-streaming no longer kicks in, command transformation is no longer performed, etc. Once the post-grace period ends, any attempt to call BSTAR functions packaged with options 1-4 causes the calling process to crash with an error message.

In order to add a license key for the product option the appropriate license update program must be run in the partition where this option is installed. The license update program adds the license key for the product option using IBM ADDLICENSE command and creates a scramble data area in QUSRSYS library. Scramble data areas

(S4F44RCS, S4F44001, S4F44002 and S4F44003) contain data unique to the IBM i(i5/OS) partition.

Warning: BSTAR release 6.3s and above have a different product system library from earlier releases and therefore can share a partition of the server with any of the earlier releases.

4. Simplified BSTAR installation

A complete BSTAR installation package for MS Windows (Windows Installer for BSTAR) can be downloaded from the web page

<http://www.cinimex.co.uk/products/bstar/predown.html>

Once the package has been installed on the personal computer, BSTAR staging application (BSTAR) is initialized. This application can then be used to distribute BSTAR products on as many IBM i(i5/OS) partitions as required directly from the PC where the application is installed. In order to use a MS Windows system to host the staging BSTAR application it is necessary to have Java runtime (JRE) installed. BSTAR installer prompts the end-user to download and install the appropriate release of JRE from the Sun Microsystems web site if no compatible JRE is found.

When BSTAR staging application is downloaded and installed from the web site, the user is prompted for the user name and password of the first IBM i(i5/OS) partition BSTAR is to be deployed on. Installation of BSTAR is automatically performed from the PC by the staging application.

To deploy BSTAR to the second and each subsequent partition BSTAR application must be used directly.

BSTAR is a Windows application and can be updated and deleted using standard MS Windows facilities.

BSTAR package contains all 5 products comprising BSTAR suite. Options that are not required can be removed using RMVLICPGM command.

The installation procedure calls QSYS2/QCMDEXC stored procedure in the target server partition. If the definition of the procedure does not exist (for example, in OS release V5R3) it is automatically created as follows:

```
CREATE PROCEDURE QSYS2/QCMDEXC(IN VARCHAR ( 32000), IN DEC (15, 5))  
LANGUAGE C SPECIFIC QSYS2/QCMDEXC NOT DETERMINISTIC MODIFIES SQL  
DATA EXTERNAL NAME QSYS/QCMDEXC PARAMETER STYLE GENERAL
```

Warning: When a new version of the operating system is installed BSTAR should also be re-installed. There is, however, no need to delete the installed BSTAR modification first; the latest modification can simply be slip-installed over the current one.

5. Thread safety and multithreaded job support

Most BSTAR BRLU functions are not thread-safe; therefore, it is not recommended to configure and use BRLU with multi-threaded jobs, such as IBM WebSphere Application Server. Also, additional threads can be created in a job defined as single-threaded by the operating system, for example, for SQL server jobs calling user-defined RPGLE functions. If a multithreaded IBM i(i5/OS) job attempts to invoke any command from CRCSSYS BSTAR system library, an error message will be generated.

BRLU makes extensive use of IBM system exit points, and some of them may be invoked in multithreaded jobs, if BRLU is installed and configured. If a BRLU exit program is invoked in a multi-threaded job, it immediately releases control causing no side effects.

6. BSTAR runtime – Option *BASE

BSTAR runtime contains common runtime functionality used by other BSTAR components.

The following runtime functions are provided:

- common runtime environment (BSTAR mode)
- checkpoint creation
- lock verification

These functions are explained at length in the Option *BASE User Guide (6.5).

6.1 System requirements

OS/400 V5R3, i5/OS V5R4, IBM i V6R1

6.2 Installation

Installation must be performed by a profile having *ALLOBJ and *SECADM special authorities. IBM RSTLICPGM command should be used with default parameter values except for the name of the licensed program, program option and save file to restore it from. The name of the distribution save file is Q4F44RCS. The following command can be therefore used to install the *BASE option:

```
RSTLICPGM LICPGM(4F44RCS)
      DEV(*SAVF)
      SAVF(QGPL/Q4F44RCS)
```

Alternatively, BSTAR runtime can be installed using the above simplified procedure (section 4).

6.3 Security settings

During the installation CRC@OWNER profile is created and CRCVENT, CRCVEXI, CRCVIN5 and CRCVCLS programs are set to adopt authorities from CRC@OWNER. [CRC@OWNER](#) default profile setting should not be changed.

BSTAR operators must also have *JOBCTL special authority.

6.4 Performance

In order to use "Backup and Recovery for Library Units" LP all database tables and data areas in the libraries of the unit must be journaled with option *BOTH to the same IBM journal known as the unit journal. Objects not journaled to the unit journal cannot be recovered. Journaling of multiple database and data area objects can significantly slow down batch processes updating the unit objects. Therefore, it is highly recommended to consider "5xxxSS1 42 OS/400 - HA Journal Performance" optional feature of the IBM Operating System. This feature, when installed and enabled for the unit journal, allows to significantly reduce processing delays caused by object journaling.

The current version of the product contains multiple performance enhancements ranging from deleting records in a file by native RPG procedure, rather than by SQL DELETE statement, to removing journal changes by parallel processes.

6.5 User Guide

6.5.1 Common runtime environment (BSTAR mode)

BSTAR mode is similar to commitment control: it can be started and ended for a specific job (STRBSTMOD and ENDBSTMOD commands). If a job already in BSTAR mode submits another job, the latter inherits the mode from its parent; it is not necessary for such processes to explicitly execute STRBSTMOD command.

BSTAR BRLU is compatible with BRLU Version 4 where the above commands had different names. For compatibility, STRBSTMOD and ENDBSTMOD commands in BSTAR system library have STRRCYMOD and ENDRCYMOD clones that can still be used.

BSTAR mode is a special kind of job environment: in this environment many standard system functions (program calls, command invocations, etc.) are intercepted and modified by BSTAR program components. Depending on the setting defined on STRBSTMOD command BSTAR can:

- intercept commands compromising journal recovery for objects (e.g. CLRPFM) and replace them with similar but "recoverable" commands

Warning: No recovery may be possible after CLRPFM command with MBR(*LAST) parameter or CPYF command with TOMBR(*ALL) parameter is executed.

RUNQRY command sending its output to a database file is only recoverable under the following conditions: OUTPUT(*OUTFILE) parameter is used and OUTFILE specifications are explicit (*RUNOPT not supported) with *MBRRPL data option, e.g. RUNQRY OUTTYPE(*OUTFILE) OUTFILE(ABCD/QRYT *FIRST *RPLMBR).

Recoverability of a library unit may be compromised by such file level operations as Sort (FMTDTA command), Initialize (INZPFM command), or DSPxxx OUTPUT(*OUTFILE) commands . If any of these commands has been issued for any file in protected libraries the file automatically becomes unprotected and has to be omitted when RMVJRNCHG recovery is executed.

Files with LOB fields are generally not recoverable using BRLU. If CLRPFM, CPYF with MBROPT(*REPLACE) or CPYFRMQRYF with MBROPT(*REPLACE) have been executed for any such file in protected libraries the file automatically becomes unprotected and has to be omitted when RMVJRNCHG recovery is executed.

- intercept SAVLIB, SAVOBJ and similar commands and replace them with equivalent save-while-active commands
- intercept any system command configured by the user for command transformation and replace it with another system command
- intercept program calls to long-running batch components and replace them with calls to equivalent multi-streamed processes

On the STRBSTMOD command one must specify the unit name and the name of control library for the unit (used by BSTAR for storing unit configuration data, e.g. CRCVCFG data area). In High Availability configurations this must be one of the replicated libraries.

All other parameters are optional. Optional parameters fall into three functional groups:

- function enablers (e.g. SAVASY and CMDRPL, enabling or disabling other BSTAR features, such as asynchronous backup or system command transformation). Functional enabler parameters are only available if related BRLU components are installed
- journal recovery enablers (JRN, LIBLIST) defining objects which may later be recovered using RMVJRNCHG command – BRLU “protects” these objects by intercepting and processing all potentially unrecoverable objects modification commands, e.g. CLRPFM
- mode modifiers (AUTOJRN, NREC, HAMODE, HOTLIB) implementing no new function but changing the way object “protection” and other functions work.

A job executing in BSTAR mode always retrieves the unit configuration from the CRCVCFG data area in *LIBL. Therefore, the control library must always be accessible via the job’s library list. STRBSTMOD command creates CRCVCFG configuration data area and CRCVYBKP checkpoint file in the control library of the unit.

Formally speaking, a job is in BSTAR mode when CRCSSYS library is above QSYS in the library list (STRBSTMOD command puts CRCSSYS library at the top of the system library list for the job) and both CRCVCFG data area containing a valid library unit configuration and CRCVERSION data area, normally residing in the BSTAR system library, CRCSSYS, are accessible via *LIBL. The library unit of the job in that case is the same as the unit defined by the first CRCVCFG data area that can be found via *LIBL.

If STRBSTMOD commands issued from multiple unit jobs (jobs working with objects in the unit libraries) define different BSTAR mode parameters, results may be unpredictable.

Before changing the control library of the unit all CLRCKPRCY command should be used to clear checkpoints and configuration objects from the current control library. Multiple CRCVCFG objects in the unit libraries can lead to unpredictable results.

If CRCSSYS library is above QSYS library in the library list of a job but CRCVCFG data area cannot be accessed via *LIBL, the job is considered to be in BSTAR mode for dummy unit. The dummy unit does not contain objects. The name of the dummy unit is @@@. This name cannot be specified on STRBSTMOD command.

ENDBSTMOD command destroys BSTAR environment in the current job. BSTAR configuration objects, however, remain in the control library of the unit.

All CRCVCFGxxx configuration objects are removed from CRCVQS library when the currently installed BSTAR release is deleted.

6.5.2 Object “protection” components

BRLU can be used for “protecting” objects in unit libraries from unrecoverable command, such as CLRPFM, CPYF OPTION(*REPLACE), etc. This protection is configured using STRBSTMOD command.

What usually happens, is CLRPFM command being replaced by a process executing in CRCV activation group and deleting all records in the target member.

Normally, delete operations are performed using activation group level commitment definition. In case of the user job running under a job level commitment definition BSTAR programs, as a rule, execute under the existing definition and issue no COMMIT or ROLLBK commands.

CLRPFM substitution command has two modes of operation controlled by NREC parameter of STRBSTMOD command, standard and parallel. Standard mode is used whenever the value of NREC parameter is *NONE or the number of records in the target file is lower than the value of NREC parameter. In other cases parallel CLRPFM mode is used. Parallel mode causes multiple jobs to be submitted to perform CLRPFM operation: each job deletes the allocated to it range of records from the target file. Parallel mode jobs always run under the commitment definition of CRCV activation group; COMMIT commands are issued for every 2000 records

deleted. In case job level commitment control has not been started standard mode makes use of the same algorithm. For jobs running using job level commitment definition CLRPFM updates to the target file are not subject to commitment control.

Warning: Some of the ILE languages (e.g. COBOL) can issue implicit CLRPFM commands at runtime (e.g. when a physical file is opened for OUTPUT). "Backup and Recovery for Library Units" tool provides no substitutes for such implicit CLRPFM commands. In the case of OUTPUT files in COBOL programs and files opened using fopen() function in C language programs it is recommended that such physical files be replaced with logical files (views). If it is undesirable to recompile the programs OVRDBF command may be used to redirect file operations to logical files (views).

If job-level commitment control status is changed (CC is started or ended) BSTAR mode for the job must be re-started (ENDBSTMOD and then STRBSTMOD).

If parallel CLRPFM is configured for the unit and the number of records in the target member exceeds 49999, then each intercepted CLRPFM command for this member is converted into a number of parallel streams with each stream (service job) deleting the allocated to it range of records. In this case CLRPFM operation is committed outside of the job commitment definition. Parallel CLRPFM jobs are submitted with CRCVCLR or, if it cannot be found in *LIBL, with *LIBL/QBATCH job description.

Starting with OS/400 V5R3, SQL DELETE processing may or may not be recoverable depending on the setting of SQL_FAST_DELETE_ROW_COUNT parameter of QAQQINI file. Obviously, if an unrecoverable DELETE statement is executed, it will be impossible to later recover the target file. There are two options available to avoid this. The first is to set SQL_FAST_DELETE_ROW_COUNT parameter to *NONE. This is a very straightforward solution, but it may have undesirable performance implications at runtime. The second is to replace SQL "DELETE * FROM table" statements with invocations of CRCVSCL stored procedure. This procedure has two parameters, library (schema) name and file (table) name, and performs exactly the same function as CLRPFM BRLU command. This command is far more efficient, especially if parallel CLRPFM is defined, than the row-by-row DELETE, and it retains all the benefits of the latter in terms of recoverability.

To be able to use CRCVSCL stored procedure it must be registered with the database manager using BRLU RGSCLRPRC command.

If all data modification operations for the unit are performed in BSTAR mode RMVJRNCHG command can later be used for recovery.

If BSTAR Program is installed REUSEDLT(*YES) setting is strongly recommended for all unit files.

A process (job) that updates any of the library unit objects (e.g. files in one of the unit libraries) while not in BSTAR mode can compromise recovery capability of the whole library unit.

6.5.3 Journal checkpoints

CRTCKPRCY command creates journal checkpoints in the journal specified on STRBSTMOD command. A checkpoint is a symbolic name given to an entry in the journal; it can later be used for journal recovery to the given entry. Checkpoints can only be created by jobs executing in BSTAR mode and having both the list of “protected” libraries and recovery journal defined by STRBSTMOD command.

6.5.4 Commands and menus

BSTAR runtime comes with BSTAR Unit Administration Menu (BSTAR command). All BSTAR system commands are available via this menu. BSTAR Administration menu is shown in the picture below.

```

BSTARA                               BSTAR Unit Administration menu
                                     System: DEMO

Select one of the following:

  1. Unit administration commands
  2. Runtime services and commands
  3. Asynchronous backup commands
  4. Command transformation commands
  5. Recovery commands
  6. Multi-streaming commands

Selection or command                   (C) Copyright Cyprolics Software 2009
===> █

F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel
F13=Information Assistant  F16=System main menu
    
```

BSTAR runtime includes the following user commands:

STRBSTMOD – to start BSTAR mode

ENDBSTMOD – to end BSTAR mode

CRTCKPRCY – to define a recovery checkpoint

For compatibility with previous releases and modifications a copy of this command named CRTCKP is automatically created in CRCSSYS BSTSR system library at product option installation time

CLRCKPRCY – to remove all previously defined checkpoints

For compatibility with previous releases and modifications a copy of this command named CLRCKP is automatically created in CRCSSYS BSTSR system library at product option installation time

CLRRCV – to remove all unused journal receivers (journal receivers containing no checkpoint entries or entries for superseded checkpoints only)

PRTCKP - to print out a list of defined checkpoints (same as CLRRCV with option *PRINT)

RGSCLRPRC – to register CRCVSCL stored procedure (CLRPFM)

DRPCLRPRC – to drop the earlier registered CRCVSCL stored procedure

6.5.5 Cleanup

BRLU product can be used to periodically clean the unit journal receivers, if the journal is defined with MNGRCV(*USER) parameter. CLRRCV command generates a printout of the currently defined checkpoints and optionally removes journal receivers that are no longer required for recovery to any of the defined checkpoints from the library unit journal receiver library.

6.5.6 Exits defined

BSTAR runtime registers the following entries for the IBM exit point QIBM_QCA_RTV_COMMAND format RTVC0100: 1872110001-1872110006. If any other product makes use of the same entries, Option *Base may either fail to install or prevent other products from being installed. In order to change the above default exit point numbers one can create a data area named CRCSEID in QUSRSYS library. The 10-character data area must contain the first number of the range to be used by 4F44RCS installation exit program.

6.5.7 “Submit concurrent batch jobs” framework

This framework is a by-product of the BRLU run-time environment. Different BSTAR functions make use of it to submit multiple concurrent batch jobs. This framework can also be used in the development environment to make the task of submitting multiple concurrent batch jobs easier.

The framework consists of two APIs, CRCVSP and CRCVSPW. CRCVSP is used to submit named groups of multiple concurrent jobs, CRCVSPW – to wait for completion of all jobs belonging to a group.

CRCVSP parameters:

1. Input CHARACTER(7) – unique identifier of the group to be created. The group is actually uniquely identified by the value of this parameter and the name of the library used to create synchronization objects (parameter 6)
2. Input DEC(3,0) – process number in the group. Process numbers must start from 1 for the first job of the group and be incremented by 1 for each following job to be submitted
3. Input CHARACTER(6000) – initial job CL command
4. Input CHARACTER(10) – name of the job description to be used (must be available via library list of the submitting job)
5. Input/Output CHARACTER(7) – effective identifier of the group to be created. It may so happen that when the first job of the new group is being submitted, jobs belonging to another group with the same name and the same synchronization library are still running. In this case the API behaves depending on the value provided in this parameter. If the initial value of the parameter is *BLKD, processing ends with an error. In any other case the API attempts to determine the first unallocated group identifier by using the first four characters of the provided unique identifier parameter as a prefix. The first unallocated identifier becomes the effective identifier of the group. The first job is then submitted and the value of the generated identifier is returned. Subsequent calls of this API to submit additional jobs in the same group must specify the value returned in this parameter as the actual name of the group (parameter 1)
6. Input CHARACTER(10) – library used to create synchronization objects. The library must exist

CRCVSPW parameters:

1. Input CHARACTER(7) – unique identifier of the group
2. Input DEC(3,0) – number of processes in the group
3. Input CHARACTER(5) – maximum time to wait for the processes to complete in minutes (*MAX special value can be used to wait forever)
4. Output CHARACTER(1): response code ('1' if errors in one or more submitted processes have been ignored, '0' otherwise)
5. Input CHARACTER(10) – name of the synchronization library

7. Asynchronous Backup and Restore – Option 1

When this option is installed and asynchronous backup configured for the library unit (STRBSTMOD command), all SAVLIB, SAVOBJ, SAV, SAVRSTLIB, SAVRSTOBJ, SAVLIBBRM, SAVOBJBRM, RSTOBJ, RSTLIB and RST commands initiated while in BSTAR mode are automatically intercepted and sent for execution to the previously started asynchronous backup processor. One such processor must be started for each unit with STRASYSAV command. The mode of executed non-save-while-active backups is changed to *SYNCLIB; SAVACTWAIT parameter value in that case is set to 900. CHKTAP commands used to unload or rewind the media, as well as DSPTAP commands, are also submitted for asynchronous execution.

The asynchronous backup processor can be ended using ENDASYSAV command. If the processor is busy servicing previously generated requests, it only ends when all outstanding requests in the queue have been processed.

All supported SAV-commands can have their command objects moved to libraries other than QSYS; redefinition of command defaults with CHGCMDDEF is also fully supported.

If objects to be saved, or tape devices, or save files, or any other objects referenced by the backup command parameters have the restricted name QTEMP, command execution mode can be forced to synchronous (synchronous commands are executed in the job that issued the original request).

All save-while-active backups are also intercepted but executed inline. The requested save-while-active mode in this case does not change. CHKTAP commands, other than those used to unload or rewind the media, are also executed inline. DSPTAP commands are executed inline if the value of OUTPUT parameter is '*'.

Asynchronous backup is available in two modes. If the SAVASY flag is set to '*YES' by STRBSTMOD command all asynchronous backup requests are executed by the backup asynchronous processor job. These backups, while asynchronous in relation to the requesting jobs, are executed in exactly the same sequence as they were requested, one by one.

This mode of operation may not be totally adequate when multiple backups can in fact be executed not only in the background, but also in parallel. Multiple backups to different save files is a good example of this. If SAVASY flag is set to '*ASYNCH' by STRBSTMOD command the asynchronous backup processor does not execute requests arriving from the request queue inline; instead, it submits those request to multiple background jobs, one request per job.

The jobs are submitted with *LIBL/CRCVSAV or, if it's not found, *LIBL/QBATCH job description. The backup processor does not in this case serialize or synchronize the backup requests in any way. In this mode synchronization must be guaranteed by the application developers. The only synchronization tool provided by BRLU product is WASYSAV command. This command, when issued from an application program, instructs the backup processor for the unit to stop accepting new

requests until all previous requests, including those submitted as independent jobs, have been completed. CHKTAP and DSPTAP command processors issue WASYSAV command under the covers.

7.1 System requirements

OS/400 V5R3, i5/OS V5R4, IBM i V6R1

BSTAR V6 Option *BASE

7.2 Installation

Installation must be performed by a profile having *ALLOBJ and *SECADM special authorities. IBM RSTLICPGM command should be used with default parameter values except for the name of the licensed program, program option and save file to restore it from. The name of the distribution save file is Q4F44001. The following command can be therefore used to install the option 1:

```
RSTLICPGM LICPGM(4F44RCS)
  DEV(*SAVF)
  OPTION(1)
  SAVF(QGPL/Q4F44001)
```

Alternatively, BSTAR asynchronous backup and restore can be installed using the simplified installation procedure (section 4).

7.3 Security settings

All background backup processors are executed under the profile of the job that submits them. Therefore, it is highly recommended to start these processors from a job running under the unit owner profile. This profile must also be authorized to perform backups of the unit libraries.

7.4 User Guide

In order for the asynchronous backup facility to be activated option 1 must be installed and SAVASY parameter of the last STRBSTMOD command issued for the unit must have the *YES value. Asynchronous Backup makes use of command message queues created, two per unit, in CRCVQS work library. The message queues in this library are permanent objects, as are the messages in the work queues. These queues should never be cleared other than by asynchronous backup server processes.

Each supported SAV-command, except for those using save-while-active mode, are intercepted and put on work queues in CRCVQS library for asynchronous execution. Each library unit has two associated work message queues: CRCVQSunt and CRCVQMunt, where unt is the name of the associated unit. The first is used to store backup requests, the second - to receive checkpoint messages.

When a backup request is intercepted it is syntax checked and placed on the request queue for the asynchronous backup server process. Control, however, does not return to the user program until the background process receives the request, submits it, and the save-while-active checkpoints are successfully taken. If the system cannot take the checkpoints an error message is returned to the user program.

The background processor executes one backup request at a time.

If the SAVASY parameter of the STRBSTMOD command is set to *ASYNCH, the asynchronous processor job accepts backup requests one by one but submits them for execution to separate jobs. These asynchronous backups are also called independent backups.

Library CRCSSYS for Option 1 contains backup command objects (e.g. SAVLIB) similar to those in QSYS library. CRCSSYS backup commands do not provide the same level of parameter validity checking.

In this configuration in order to change system command defaults it is sufficient to change SAVxxxxxx command objects only in QSYS library. Likewise, all previously redefined command defaults are not effectively changed by Backup for Library Units LP.

7.4.1 Commands and menus

This option has four user commands:

STRSAVASY - for starting the background processor

DSPSAVRQS - for displaying the currently executing backup requests

DSPSAVRQA - for displaying the currently executing independent backup requests

WASYSAV - to instruct the active backup processor that no new requests can be accepted until all previously issued requests have been completed.

All these commands have one parameter: the name of the library unit.

DSPSAVRQS command displays outstanding SAV-requests and the status of the submitted background processor. 'RRCV' status means that the server process is waiting for a new request. 'RRCA' status has the same meaning as 'RRCV' but additionally indicates that there are unfinished independent backup requests (executed independently, not by the background processor job). 'MSGW' status means that the server process is waiting for manual intervention. All other status codes are the same as you would see on the WRKACTJOB display for a job.

DSPSAVRQS can optionally clear the request message queue deleting all currently pending requests. The monitor screen it displays looks like this:

```

                                BSTAR asynchronous backup processor
Unit: TST                               2/09/09 09:37:39
Refresh in seconds . . . . . 015
Server job . . . . . 077128/CINOWNER/SAVJOBST
Server job state . . . . . Active/RRCA/INDEP

Type options, press Enter.
  1=Display request  4=Remove request  5=Work with job

                                Pending backup requests

Opt Date  Time  Job      User      Number Command
-----
█ 020909 093653 QPADEV000C CINOWNER  077127 QSYS/SAVLIB LIB (V38BASELIB) DEV (

Enter=refresh PF1=help PF2=indep.requests PF12=exit PF19=auto-refresh off
    
```

The screen is automatically refreshed. It is possible to change the frequency of refresh operations by changing the related DSPSAVRQS parameter.

When independent backup mode is defined as part of BSTAR mode, some of the requests in the queue may still be synchronous, i.e. executed by the background processor job. For example, “Wait for asynchronous backups” request generated by WASYSAV API is always executed by the background processor in synchronous mode. Additional information about independent requests can be obtained by pressing F2 from the main background processor status display. Asynchronous backup monitor screen is auto-refreshed at the end of the interval specified on the command REFRESH parameter. The default interval is 15 seconds.

All BSTAR asynchronous backup and restore system commands are available via RCVR menu shown in the picture below.

```

ABCKP                               BSTAR Asynchronous Backup Menu
                                     System: DEMO

Select one of the following:

  1. Start asynchronous backup processor
  2. End asynchronous backup processor
  3. Display backup requests in queue
  4. Wait for backup requests in queue

Selection or command                  (C) Copyright Cyprolics Software 2009
===> █

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel
F13=Information Assistant   F16=System main menu

```

7.4.2 Background backup server

The backup server job for the 'unt' unit has SAVJOBunt name and is submitted using *LIBL/CRCVJOB job description. If this job description is missing, *LIBL/QBATCH job description in the same library is used. Whatever job description is used to submit the server job, it is advisable to use default replies to request messages (CHGJOB INQMSGRPY(*DFT)). Otherwise, the background job might often require manual intervention (e.g. in case data is being saved into a non-empty save file).

To stop the background server process ENDJOB OPTION(*CTRLD) DELAY(600) command should be issued.

7.4.3 Exits defined

Option 1 registers the following entries for the IBM exit point QIBM_QCA_RTV_COMMAND format RTVC0100: 1872110011-1872110022.

If any other product makes use of the same entries, Option 2 may either fail to install or prevent other products from being installed. In order to change the above default exit point numbers one can create a data area named CRCSEID in QUSRSYS library. This data area must be preset the way it was described in the Option 1 Exits section.

8. BSTAR Command Transformation – Option 2

This option enables definition of backup command transformation configuration where IBM i(i5/OS) system commands can be transformed or replaced at runtime.

There are two transformation mechanisms provided. Although they are very similar in function, implementation details are not exactly the same. The first type of transformation applies to the following commands:

SAVLIB, SAVOBJ, SAV, RSTLIB, RSTOBJ, RST, CHKTAP, SAVOBJBRM, SABLIBBRM, and DSPTAP

Transformation of this type is only available when BSTAR Option 1 is installed. To enable the transformation STRRCYMOD command must be executed with CMDRPL(*YES) parameter.

The above backup and recovery commands can be replaced with any system commands (e.g. SAVLIB - with SAVLIBBRM, SAVOBJ to tape - with SAVOBJ to a save file, etc.). This feature allows an old system based on SAVLIB/SAVOBJ backups to use new backup facilities.

Transformation of the second type is called odd transformation. To enable transformation of system commands (see the description of QIBM_QCA_CHG_COMMAND exit point for limitations) this transformation must be explicitly enabled for any target command object.

8.1 System requirements

OS/400 V5R3, i5/OS V5R4, IBM i V6R1

BSTAR V6 Option *BASE

8.2 Installation

Installation must be performed by a profile having *ALLOBJ and *SECADM special authorities. IBM RSTLICPGM command should be used with default parameter values except for the name of the licensed program, program option and save file to restore it from. The name of the distribution save file is Q4F44002. The following command can be therefore used to install the option 2:

```
RSTLICPGM LICPGM(4F44RCS)
  DEV(*SAVF)
  OPTION(2)
  SAVF(QGPL/Q4F44002)
```

Alternatively, BSTAR command transformation can be installed using the simplified installation procedure (section 4).

When a new release of BSTAR is installed over the existing 6.Xs release the existing odd command transformation settings (see section 8.4.3) are preserved and are automatically enabled in the new release. When a new release of BSTAR is installed over an older release that release is not removed from the system; the existing odd transformation definitions are not automatically imported into the repository of the new release and have, therefore, to be manually redefined.

8.3 Security and audit settings

Mainly same as for *BASE option. If QAUDTCL system value list contains *OBJAUD setting BSTAR generates an audit trail record for each successful command transformation. The entry is sent to QAUDJRN system audit journal and has the entry code '4F'.

8.4 User Guide

Whatever the transformation type (automatically enabled transformation for Option 1 backup and recovery commands included in Option 1 and 2 of BSTAR, or odd transformation), transformation details for a unit can be defined in the transformation source file. This file must be defined with record length of 112 bytes. In order to locate this file BSTAR uses the following search order:

- if CRCVCMDunt data area, where unt is the name of the unit, can be found in CRCVQS library, then the first 10 characters of it are treated as the library name and the second 10 – as the file name
- If the data area and the file pointed to by the data area exist, the latter is used as the transformation definition source file for the unit
- If either the data area or the file does not exist, BSTAR checks CRCVCMDunt file in CRCVQS library. This file, in case it exists, is used as the transformation definition source file for the unit
- If the above procedure does not locate a valid source file, the last attempt is made to find the definition source file under the name CRCVCMD in CRCVQS library. In case this file does not exist command transformation facility is disabled

A sample CRCVCMD file is created in CRCSSYS library by the Option 2 installation program. In order to create a new command transformation file CRCVCMD sample file has to be copied into the appropriate library using CRTTFMFIL command.

To substitute a backup command executed in BSTAR mode this command with all its runtime parameters must be defined in the source section of a CRCVCMDunt definition member. If, for instance, at runtime SAVLIB LIB(KFILPRO) DEV(TAP01) TGTRLS(V5R3) command is issued and it is desirable to modify it before execution, a new replacement definition must be created with the source section thereof containing exactly this command string with the /**/ separator in the first four positions of the last line. Multiple blanks and single '+' signs are removed at runtime before the command issued by the application is compared with the source string. Blanks, however, are not removed from literals. If literals are used in the comparison template, they cannot span multiple lines. The command replacement

member must have the name prefix of the command being substituted. The target section of the definition has the same structure as the source section and can define any other backup command. If option 1 of BSTAR is also installed and activated, commands after transformation can be sent to the asynchronous backup processor for execution.

Wildcards (_ - underscore) are allowed in transformation definitions. A wildcard character stands for any single character. Wildcards, however, are not allowed in command names.

Also, the following substitution variables can be defined: @UN (same as &&& or &01) is replaced by the command transformation processor with the name of the unit, and @BK (same as &03) – with the contents of *CHAR(3) CRCVBKPID data area, if it is found in *LIBL, @PRDAT – with the current processing date value in the same format as RTVSYSVAL SYSVAL(QDATE) would return.

The members of CRCVCMD source file are processed in alphabetical order, and when a match is found search is terminated.

It is possible to condition transformation defined in each member of CRCVCMD file by using the member description field. If the first three characters of this field contain 'IF:' (case sensitive) then the rest of the field can be used to define CL-compatible logical expression containing substitution variables [&UN](#), &EP, &BK, and &PRDAT. In order to enable the conditions they must be compiled using CMPCND command. If any conditions change, CMPCND compilation has to be used again in order to refresh the earlier compiled condition implementation programs.

A condition implementation program is a CL program having the same name as the file it is created for and residing in the same library as the file.

8.4.1 Example

To substitute SAVLIB LIB(KFILPRO) DEV(TAP01) TGTRLS(V5R3) and SAVLIB LIB(KFILPRO) DEV(TAP01) TGTRLS(V5R3) commands with

```
SAVLIBBRM LIB(KFILPRO) DEV(*NONE) MEDPCY(*NONE)
    TGTRLS(V5R3M0) DTACPR(*YES) EXPDATE(*PERM)
    MOVPCY(*NONE) MEDCLS(SAVSYS) LOC(*ANY)
    SAVF(*YES) SAVFASP(1) SAVFEXP(*NONE)
    MAXSTG(1) VOLSEC(*NO) MINVOL(1) MARKDUP(*NO) command,
```

SAVLIB1 member containing the following definition can be added to CRCVCMDunt file in CRCVQS:

```
-----
SAVLIB LIB(KFIL@UN) DEV(TAP01) +
    TGTRLS(V5R___)
/**/
SAVLIBBRM LIB(KFIL@UN) DEV(*NONE) MEDPCY(*NONE) +
    TGTRLS(V5R3M0) DTACPR(*YES) EXPDATE(*PERM) +
```

```

MOVPCY(*NONE) MEDCLS(SAVSYS) LOC(*ANY) +
SAVF(*YES) SAVFASP(1) SAVFEXP(*NONE) +
MAXSTG(1) VOLSEC(*NO) MINVOL(1) MARKDUP(*NO)
/
```

Neither of the sections may exceed 900 lines, excluding the separator line.

The target part of the definition can be empty. For example,

```

-----
CHKTAP DEV(TAP01)
/
```

definition would cause CHKTAP DEV(TAP01) commands to be ignored.

In order to be processed asynchronously commands must be supported by Option 1 and be issued with no qualifiers. For example, neither QSYS/SAVOBJ, nor CALL PGM(ABCD) commands can be sent to the background processor for execution, even if the match is found and they are selected as replacement commands. However, if ABCD program contains other backup commands, they can be executed asynchronously.

It is not always obvious what backup, restore or other parameters are used by the application when SAVLIB, SAVOBJ, RSTLIB, RSTOBJ, CHKTAP, DSPTAP or other commands are issued. If CMDRPL parameter value of STRBSTMOD command is set to *SOURCE, command strings for all of the above commands and all commands with odd transformation enabled (see 8.4.3) issued by application while in BSTAR mode are stored as messages in CRCVCMDunt message queue in CRCVQS library. Information in these messages can then be used to define command transformations. Since some of the command parameters are modified by BSTAR BRLU and other programs before execution, it is highly recommended to use the contents of the CRCVCMDunt message queue as the only source of command source data.

8.4.2 Parsed substitution variables

In addition to the above-described simple substitution variables there exist so-called parsed substitution variables. There are five variables of this type defined: @P0-@P9. Each of them has the CHARVAR(20) type. These variables are assigned values when the pattern of an issued CL command matches the pattern of the command transformation source string defined in CRCVCMDunt file. These variables can therefore be mapped to certain sub-strings of the CL command string.

The transformation definition from 8.4.1 can be modified as follows:

```

-----
SAVLIB LIB(@P1) DEV(TAP01) +
  TGTRLS(V5R_)
/**/
  SAVLIBBRM LIB(@P1) DEV(*NONE) MEDPCY(*NONE) +
    TGTRLS(V5R3M0) DTACPR(*YES) EXPDATE(*PERM) +
    MOVPCY(*NONE) MEDCLS(SAVSYS) LOC(*ANY) +
    SAVF(*YES) SAVFASP(1) SAVFEXP(*NONE) +
    MAXSTG(1) VOLSEC(*NO) MINVOL(1) MARKDUP(*NO)
/**/
-----

```

If SAVLIB LIB(ABCD) DEV(TAP01) TGTRLS(V5R3M0) command is then issued, BSTAR transformation engine will be able to match the first 12 characters of this command with the first 12 characters of the source definition string and, using parsing operation, will assign sub-string starting at the position of the parsed variable in the source definition string and ending before the next separator character ('), '(', '/', ' ', and ''). In this example @P1 will be assigned the value "ABCD". This library will then be backed up using BRMS.

8.4.3 Odd transformation

While transformation for each of the backup and recovery commands included in Options 1 and 2 of BRLU is automatically enabled when STRBSTMOD command is issued for a unit with CMDRPL(*YES) parameter, transformation for other system commands (e.g. DSPLIB) must be explicitly enabled for each such command. In order to do this, ENAODDTFM command must be executed for any such command object. To enable the transformation mechanism QIBM_QCA_CHG_COMMAND exit point is registered for the command object. Only 100 odd commands can be enabled for transformation at any point in time. The entries with numbers from the 1872110101-1872110200 range are used by default. CRCSEID data area can be used to redefine the range.

Once transformation has been enabled for a new command the actual transformation rules can be defined in the CRCVCMD file for the library unit.

DISCMDRPL disables command transformation for the given command object.

Command transformation should not be defined for any commands in CRCSSYS library.

When Option 2 is deleted all previously defined QIBM_QCA_CHG_COMMAND exit points are removed from the system. It is advisable to create a CL program containing ENAODDTFM commands for the required system command objects and execute it each time the current release of Option 2 is deleted and a new release is reinstalled.

Command transformation in the dummy unit (@@@) cannot be configured.

8.4.4 Commands and menus

BSTAR commands command transformation commands:

ENAODDTFM – enables odd command transformation

Warning: Transformation for some of the IBM i(i5/OS) system commands cannot be enabled. The list of recognized exceptions in the current release includes the following commands: DSPLICKEY, CRTMSGQ, CHGJRN, CRTJRN, CRTJRNRCV, DLTJRNRCV, RTVNETA, DLTCMD, RTVSYVAL, RTVDTAARA, SNDPGMMMSG, RTVOBJD, CHKOBJ, HLDJOB, ALCOBJ, DLCOBJ, DSPSPLF, STRJRNPF, CHKPRDOPT, STRQMQRV, RCVF, and RTVJOBA. If ENAODDTFM command is used to enable transformation for any of the “forbidden” command, it will return an error message.

Should transformation for any such command be a strong requirement, the appropriate Request for Modification (RFM) can be raised via Cinimex Support channel.

If transformation has to be defined for system commands having their clones in CRCSSYS library, transformation should be defined for command objects in CRCSSYS, unless command references are qualified, e.g. QSYS/CPYF, in which case transformation for the command in QSYS can be defined, but the source string has to be entered with the appropriate qualifier.

DISCMDTFM – disables previously defined odd command transformation

CMPCND – enables (refreshes) conditional processing of transformation definitions

LSTODDTFM – lists system commands with odd transformation enabled

CRTTFMFIL – creates a new command transformation definition source file from template

WRKTFMDFN – invokes WRKMBRPDM command to display members of the file earlier defined as the transformation source file for the unit using CRTTFMFIL command.

CRTTFMFIL command both creates the new source file and links it to the unit using pointer data area CRCVCMDxxx in CRCVQS. Therefore, it is highly recommended to use CRTTFMFIL command to define the transformation source file for each unit. If the file with the given name already exists, CRTTFMFIL simply links it to the unit but does not update it.

Transformation can be dynamically switched on and of using CMDRPL parameter of STRBSTMOD command. The parameter takes immediate effect.

All BSTAR Option 2 system commands are available via CMDSB menu shown in the picture below.

```

CMDSB                               BSTAR Command Transformation Menu
                                     System: DEMO

Select one of the following:

  1. Enable odd command transformation
  2. Disable odd command transformation
  3. Compile conditional transformation
  4. List enabled odd transformations
  5. Work with transformation definitions
  6. Create definition file from template

Selection or command                   (C) Copyright Cyprolics Software 2009
===> █

F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel
F13=Information Assistant  F16=System main menu
    
```

8.4.5 Compatibility

BSTAR Option 2 is based on the command replacement engine included in 4F44RCS *BASE option.

If Option 2 is installed but Option 1 is not, command replacement for backup commands (packaged in Option 1) must be defined explicitly. For that, ENAODDTFM command must be issued for all related command objects.

8.4.6 Exits defined

Option 2 registers the following entries for the IBM exit point QIBM_QCA_RTV_COMMAND format RTVC0100: 1872110031,1872110032, 1872110033. Odd command transformation facility optionally makes use of exit points starting from 1872110101. If any other product makes use of the same entries, Option 2 may either fail to install or prevent other products from being installed. In order to change the above default exit point numbers one can create a data area named CRCSEID in QUSRSYS library. This data area must be preset the way it was described in the Option 1 Exits section.

8.4.7 Service trace

Command transformation configurations can be very sophisticated, but even a simple typo in the definition can prevent the transformation function from working properly. Moreover, these typos can be extremely hard to find. In order to find out why configured transformations do not work as intended Option 2 service trace facility can be used. Service trace is activated automatically if the job is placed in BSTAR mode with command transformation enabled and if message queue CRCOPT2 is available in *LIBL. Transformation trace messages are sent to this message queue and can then be printed using DSPMSG system command. Although command transformation service trace records can only be fully analyzed by BSTAR service representatives, some of the data in the trace is self-explanatory and can be used for trouble-shooting.

9. Recovery for Library Units – Option 3

This option provides “recovery-to-checkpoint” functionality for files and data areas in library units. There are two different interfaces to the recovery function. The first is low-level, where all recovery parameters, such as names of libraries to be recovered, number of recovery streams, etc. must be explicitly defined using RCYTOCKP command parameters. The other is high-level; using this high-level interface it is possible to define standing recovery data for a library unit and then simply refer to it at the time of recovery. Therefore, important decisions, such as how many recovery streams to use, what files to omit, and what files to recover in separate streams, can be taken at the system configuration, rather than recovery time.

Option 3 can be used to create IBM i(i5/OS) work management objects, such as job descriptions, job queues and subsystems, to improve performance of the recovery process. Management of these objects, including starting and stopping of the recovery subsystem, can be totally transparent to the operator performing the recovery.

BSTAR Recovery for Library Units supports multi-streamed recovery when multiple jobs are submitted to share recovery workload.

9.1 System requirements

OS/400 V5R3, i5/OS V5R4, IBM i V6R1

BSTAR V6 Option *BASE

9.2 Installation

Installation must be performed by a profile having *ALLOBJ and *SECADM special authorities. IBM RSTLICPGM command should be used with default parameter values except for the name of the licensed program, program option and save file to restore it from. The name of the distribution save file is Q4F44003. The following command can be therefore used to install the option 3:

```
RSTLICPGM LICPGM(4F44RCS)
  DEV(*SAVF)
  OPTION(3)
  SAVF(QGPL/Q4F44003)
```

Alternatively, BSTAR recovery for library units can be installed using the simplified installation procedure (section 4).

9.3 Security settings

Same as for *BASE option.

9.4 User Guide

All objects (database tables and data areas) to be recovered by BSTAR must be journaled with option *BOTH to the same IBM journal. Obviously, journaling for the objects has to be started before updates begin. BSTAR includes support for *MAXOPT3 journals (large journal sequence numbers).

BSTAR Recovery for Library Units is based on the concept of checkpoints. Checkpoints are system objects created at runtime that can later be referenced at the time of recovery. A checkpoint can be used to recover a single library unit.

Consistent recovery can only be achieved if all IBM i(i5/OS) processes updating objects from a certain library unit execute in BSTAR mode.

To create a checkpoint Create Checkpoint for Recovery(CRTCKPRCY) command must be used. This command can only be issued if the job issuing the command is already in BSTAR mode (native or inherited). A checkpoint is normally taken at the moment of no system activity.

If a job issuing CRTCKPRCY mode is not in BSTAR mode but if STRBSTMOD was issued for the unit in the past, CRTCKPRCY makes use of the STRBSTMOD parameters stored in CRCVCFG data area to implicitly start BSTAR mode for the job.

Special checkpoint name @LAST is reserved for the latest checkpoint taken for the unit.

Two types of recovery are defined: *NEW and *CONT. Recovery of the type *CONT is used when the already recovered unit has to be rolled back even further – to one of the earlier checkpoints. If two checkpoints A and B have been defined for a unit, the following BSTAR recovery scenario is possible:

- Step 1. Recover to checkpoint *LAST (i.e. B) using recovery identifier ONE
- Step 2. Re-enter BSTAR mode
- Step 3. Define checkpoint C
- Step 3. Modify unit data
- Step 4. Recover to checkpoint @LAST (this time it is C) – this recovery rolls unit objects back to their checkpoint B state
- Step 5. Recover to checkpoint A using recovery identifier ONE and recovery type *CONT

9.4.1 Recovery in High Availability Environments

The only high availability software currently supported is MIMIX.

In order for the Recovery for Library Units Licensed Program to successfully operate in the MIMIX environment the object and the file entry for CRCVYBKP multi-member file must be added to the primary group definition. Further on, after STRRCYMOD command is executed for the first time for a library unit, it is strongly recommended to make sure that CRCVYBKP file has been created in the first unit library on both the source and the target systems, and that journaling for this file has been successfully started.

There are two possible approaches to recovery in HA environments. Both of them assume that CRTCKPCRY commands used to define checkpoints were run with HAMODE(Y) parameter (CRTCKPCRY command, if issued with CFG(*IMPLICIT) parameter, refers to the HA setting of the previous CRTCKPCRY command for the same unit; if the first ever CRTCKPCRY command for the unit is executed with CFG(*IMPLICIT) parameter, HAMODE defaults to 'N' and HA mode is, therefore, not used).

The first approach is more universal but requires additional manual effort. The second is almost seamlessly coupled with MIMIX, but involves certain manipulations of the MIMIX replication facility.

Approach 1.

MIMIX replication is suspended (DG is stopped) and RCYTOCKP commands are executed independently for both the production and the backup units; after that MIMIX replication is cold-started. This approach has a pre-requisite of the HA data group to be defined with journaling on the target system for both files and data areas.

Approach 2.

Prior to starting the EOD MIMIX HLDDGLOG command is executed for the replication data group. If EOD process fails, MIMIX replication is not suspended for the data group and recovery is executed only for the live unit. MIMIX replicates the changes to the backup system. After the EOD process ends RLSDGLOG command is executed for the data group.

As RCYTOCKP process ends journaling of CRCVYBKP file, it may cause MIMIX warning messages to be issued in case Approach 2 is used.

Whatever method is used, once the recovery process ends journaling for CRCVYBKP file in the control library must be manually restarted, as it is also stopped by the recovery process.

To make the above synchronous recovery in HA environment possible CRTCKPCRY command, when executed with HAMODE(*YES) parameter, waits for MIMIX backlog to disappear before taking the checkpoint. If this represents a serious problem, CRTCKPCRY in HA environments can be issued with HAMODE(*NO) parameter. In such cases, however, only the production side of the library unit can be recovered to thus defined checkpoints. MIMIX synchronization can be attempted later.

9.4.2 Menu

Option 3 facilities can be accessed via BSTARR system menu (GO CRCSSYS/BSTARR).

```

BSTARR                               BSTAR Unit Recovery Menu
                                     System: DEMO

Select one of the following:

  1. Enable advanced recovery for unit
  2. Advanced recovery to checkpoint
  3. Delete unit configuration objects
  4. Manual recovery to checkpoint

  7. Create checkpoint

 11. BSTAR main administration menu (BSTARA)

Selection or command                  (C) Copyright Cyprolics Software 2009
===> █

F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel
F13=Information Assistant  F16=System main menu
    
```

Option 1 “Enable advanced recovery for unit” creates BSTAR standing data for the library unit. In order to do this BSTAR runs the user through a mock unit recovery scenario executing STRBSTMOD, IVJRN, CRTCKPRCY, VFYFDLCK, and RCYTOCKP commands in prompt mode. The entered parameters are stored in work structures in CRCVQS library. It is possible to make the system skip VFYFDLCK step during the advanced recovery by setting LIBLIST parameter value to *NONE on VFYFDLCK command during the mock recovery process.

Also, work management structures for the recovery process can be created. The list of such work management objects includes CRCVSBSunt subsystem, CRCVSBSunt job queue, and CRCVRJDxx job descriptions; the subsystem description and the job queue are saved in CRCVQS library, and the job descriptions – in the control library for the unit. The subsystem has 10 main storage pools defined, *SHRPOOL1-*SHRPOOL10. The number of job descriptions created is max(n+1,10) where n is the value of the “additional file streams” (FLIST) parameter of RCYTOCKP command. At execution time the first max(n+1,10) recovery streams are, therefore, run in separate main storage pools. Additional job CRCVRJDxx job descriptions may be added as required. CRCVSBSunt subsystem description may have to be changed accordingly. The tool, however, exercises no control over the actual main storage allocation. The recommended strategy is to use QPFRADJ=3 setting thus allowing the operating system to automatically allocate main storage to shared pools.

Advanced recovery configuration has to be redefined each time a new release of BSTAR is installed.

Option 2 “Advanced recovery to checkpoint” performs simplified recovery using VFYFDLCK and RCYTOCKP commands. To execute RCYTOCKP its parameters are extracted from the earlier standing data structures. There are two sub-options available: prompted recovery, when the end user can change some of the earlier defined parameters and has to press Enter key in order to let the process run, and automatic recovery, when the related commands are executed with no user intervention.

The earlier defined recovery subsystem for the unit can be started and ended automatically by the above recovery process.

Option 3 “Delete unit configuration objects” deletes all unit related configuration objects from CRCVQS library. The only exception is CRCVCMDunt command transformation definition file that has to be deleted manually.

Option 4 is the interface to the low-level unit recovery interface, RCYTOCKP command. Most of its parameters are self-explanatory; help text is also provided. A few general comments, however, should be made.

Number of recovery streams parameter is used to define the MINIMU number of recovery streams requested. The actual number is calculated on the basis of the number of objects to be recovered and explicitly specified additional recovery streams. Additional recovery streams are the first to be submitted; their numbering starts from 1. This can be used to send them to different job queues and, ultimately, different memory pools for execution.

*OBJHLD object type in the ROLIST parameter stands for “object holder”. Object holder is a file having the format generated by DSPOBJD DETAIL(*BASIC) system command. Such files are not recovered themselves; they are used to extract the names of objects to be recovered. The value of LIBLIST parameter in this case has to be set to *OBJECTS. This feature can be used to selective recovery, especially when the number of the files to be omitted from the recovery process is too high for them to be specified using OFLIST parameter.

9.4.3 Commands

BSTAR recovery for library units commands:

ENARCYUNT – to run the mock recovery process and create BRLU recovery standing data

RCYTOCKP – to recover to checkpoint

RCYTOCKPA – to invoke advanced recovery for library unit

DLTUNTCF – to delete unit configuration objects from CRCVQS library

IVJRN – to verify and/or start journaling for the library unit objects. It is recommended to use this command before entering BSTAR mode for the unit for the first time. IVJRN analyses and, if journaling characteristics for an object are not

correct, restarts journaling for all of the library unit objects thus making sure that journaling for the unit is consistent and complete

VFYFDLCK – to verify existing locks for physical files and data areas. It is strongly recommended to run this verification before each recovery operation (RCYTOCKP). For physical files, however, only locks for *FIRST members are verified. Locks held by the job where VFYFDLCK command is run are not verified. If a lock is detected, the related object is displayed using WRKOBJLCK command (this only happens if the command is running in an interactive job). When the lock is released, pressing the “Enter” key causes lock analysis to continue. If locks for the displayed object still exist, the processing will end with an error message.

10. Help panels

BSTAR command help panels contain more detailed information regarding the command parameters and the mode of execution.