

Cypriolies Ltd.

iSTREAM CL Command Transformer (CCT) Guide

Version 4.2.0.0

Table of Contents

| | | |
|-----------|--|-----------|
| 1. | Introduction to iSTREAM CCT | 5 |
| 2. | iSTREAM runtime – Option *BASE | 6 |
| 2.1 | Introduction | 6 |
| 2.2 | Common runtime environment (iSTREAM mode) | 6 |
| 2.3 | Common unit (@@) | 8 |
| 2.4 | Introduction to interfaces, commands and menus | 9 |
| 2.5 | Proxi command objects and chain transformations | 10 |
| 2.6 | iSTREAM APIs | 11 |
| 2.6.1 | <i>"Submit concurrent batch jobs" framework</i> | <i>11</i> |
| 2.6.2 | <i>Other APIs</i> | <i>12</i> |
| 3. | Flash Operations – Option 1 | 13 |
| 3.1 | Introduction | 13 |
| 3.2 | Configuration and function | 14 |
| 3.3 | Remote Flash option | 16 |
| 3.3.1 | <i>Configuration and Operation</i> | <i>16</i> |
| 3.3.2 | <i>Network configuration</i> | <i>17</i> |
| 3.3.3 | <i>Security settings</i> | <i>17</i> |
| 3.4 | Commands and menus | 18 |
| 3.5 | CPYLIBF command | 20 |
| 3.6 | SBMICMD command | 20 |
| 3.7 | SAVACT option override | 20 |
| 3.8 | Exits defined..... | 20 |
| 3.9 | Service trace | 21 |
| 4. | iSTREAM Command Transformation – Option 2 | 22 |
| 4.1 | Introduction | 22 |
| 4.2 | Configuration and function | 22 |
| 4.2.1 | <i>Functional description</i> | <i>22</i> |
| 4.2.2 | <i>Command Transformation Example</i> | <i>24</i> |
| 4.2.3 | <i>Parsed substitution variables</i> | <i>26</i> |
| 4.2.4 | <i>Partial matching</i> | <i>26</i> |
| 4.2.5 | <i>CKSTACK specification</i> | <i>27</i> |
| 4.2.6 | <i>Control specification sequence</i> | <i>28</i> |
| 4.2.7 | <i>Prefix "++"</i> | <i>28</i> |
| 4.2.8 | <i>CPYF multistreaming transformation</i> | <i>29</i> |
| 4.2.9 | <i>Transformation enablement</i> | <i>30</i> |
| 4.3 | Substitution variable name modifier | 30 |
| 4.4 | Commands and menus | 31 |
| 4.5 | Exits defined..... | 32 |
| 4.6 | Audit trail..... | 32 |
| 4.7 | Service trace | 32 |
| 5. | Rollback for Library Units – Option 3..... | 34 |
| 5.1 | Introduction. Object protection | 34 |
| 5.1.1 | <i>Object "protection" components</i> | <i>34</i> |
| 5.1.2 | <i>Object "protection" setup</i> | <i>35</i> |
| 5.1.3 | <i>Journal checkpoints</i> | <i>37</i> |
| 5.1.4 | <i>Cleanup</i> | <i>37</i> |
| 5.1.5 | <i>Reporting</i> | <i>37</i> |
| 5.1.6 | <i>Clone command option</i> | <i>37</i> |
| 5.2 | Rollback interfaces | 39 |
| 5.3 | Rollback configuration and function | 39 |
| 5.3.1 | <i>Functional description</i> | <i>39</i> |



iSTREAM Command Transformer (CCT) Guide

| | | |
|-----------|---|-----------|
| 5.3.2 | <i>Parallel multithreaded rollback</i> | 41 |
| 5.3.3 | <i>Rollback in High Availability Environments</i> | 43 |
| 5.4 | Menus and commands | 45 |
| 5.5 | Exits defined..... | 49 |
| 6. | CL command transformation algorithm | 50 |
| 7. | iSTREAM CCT restrictions | 52 |
| | Appendix A. STRISTMOD command notes | 53 |

Trademarks

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



1. Introduction to iSTREAM CCT

iSTREAM is a Licensed Program for IBM i operating system designed to improve performance characteristics of long running batch processes. Performance improvements are generally achieved by implementation of multiple parallel programming techniques. iSTREAM consists of four main components, Performance Investigator, also known as iSTREAM PI, iSTREAM Accelerator, iSTREAM File Replicator, and a Graphical User Interface in the form of iSTREAM Access for MS Windows. iSTREAM Accelerator includes two subcomponents, iSTREAM CL Command Transformer(CCT) and iSTREAM Generic Multistreaming Toolkit.

iSTREAM is packaged as an IBM i Licensed Product (7S77STR). The complete list of all iSTREAM options, their installation procedures, system requirements, security and other similar subjects can be found in iSTREAM Planning and Installation Guide.

This manual contains a functional description of options *BASE, 1, 2 and 3 of iSTREAM. Option *BASE represents iSTREAM common runtime environment; options 1, 2 and 3 include functions of iSTREAM CL Command Transformer (CCT).

iSTREAM CCT is primarily designed to improve performance characteristics of different CL commands traditionally executed as part of long running batch applications.

Option 1 (Flash Operations) of CCT supports definition of the flash mode of execution for selected CL commands. In this mode, specified commands are intercepted and submitted for flash (asynchronous) execution to the background. Commands of the SAV group, e.g. SAV, SAVLIB, SAVOBJ, etc., are also enriched with save-while-active parameters letting the main execution thread of the batch job proceed to the next function without waiting for backup operations to complete. This, in effect, turns on the flash mode for such commands.

Option 2 (Command Transformation) can be used to define any runtime transformation of almost any CL command (exceptions are those commands that QIBM_QCA_CHG_COMMAND exit would not support). This facility can be used, for example, to improve performance of CPYLIB and CPYF commands by transforming them into functionally equivalent SAVRSTLIB and SAVRATOBJ commands.

Option 3 offers a new approach to data recovery. This function can be used to enable "rollback by system journal" thus setting up an alternative recovery policy for the transformed batch processes.

The maximum length of command strings that can be processed by iSTREAM is 5000 characters.

2. iSTREAM runtime – Option *BASE

2.1 Introduction

A library unit is a group of IBM i libraries (program and data) representing a logical copy of an application.

iSTREAM runtime contains common functionality used by other iSTREAM components.

Warning: option *BASE of iSTREAM is rarely used on its own. Although it does include common product components, specific CCT functions require the appropriate functional options of the iSTREAM product to also be installed. This section, however, by way of introduction, focuses on the general runtime environment per se. For a more detailed discussion of specific CCT functions see information in sections related to the appropriate product options.

2.2 Common runtime environment (iSTREAM mode)

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

iSTREAM 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 iSTREAM program components. Depending on the setting defined on STRISTMOD command, iSTREAM can:

- intercept CL commands compromising journal rollback for objects in libraries, e.g. CLRPFM, and transform them into similar but “recoverable” commands
- intercept SAVLIB, SAVOBJ and similar CL commands, transform them into equivalent save-while-active commands and submit to batch for execution
- intercept any CL system commands configured by the user for explicit command transformation and replace them with other system commands
- intercept program calls to long-running batch components and replace them with calls to equivalent multistreamed processes (this capability is described in detail in iSTREAM Generic Multistreaming Toolkit manual).

On the STRISTMOD command one must only specify a unit name. If Rollback functionality (Option 3) is used, the other mandatory elements are the name of control library for the unit (used by iSTREAM for storing unit rollback configuration



data) and the name/library of journal object. 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. ASYEXE and CMDTFM, enabling or disabling other iSTREAM features, such as flash CL command execution or CL system command transformation). Functional enabler parameters only become available if related CCT options are installed
- journal rollback enablers (JRN, LIBLIST) defining objects which may later be rolled back using RMVJRNCHG command – CCT “protects” these objects by intercepting and processing all potentially unrecoverable object modifications by CL commands, e.g. CLRPFM
- mode modifiers (AUTOJRN, NREC, HOTLIB, WLCASY) implementing no new functions but changing the way object “protection” and other functions work

It is possible to define all the required iSTREAM mode parameters using STRISTMOD command and then, at runtime, use STRISTMUD command that define the iSTREAM mode for the current job with all the default parameter for the unit entered earlier.

A job executing in iSTREAM mode has a job-level environment variable `ISTREAM_UNIT` set to the name of the current library unit. The unit configuration is stored in `CRCVCFGxxx` data are in `ISTVQS` library, where `xxx` is the name of the unit. Under certain condition STRISTMOD command also creates a checkpoint file in the control library of the unit. The name of the file has the format `ISTVCKPunt`, where `unt` is the name of the library unit.

Formally speaking, a job is considered to be in iSTREAM mode when `ISTSSYS` library is in the library list (STRISTMOD/STRISTMUD adds `ISTSSYS` library to either the system or user portion of the library list for the job), the environment variable with the name `ISTREAM_UNIT` is available and its value is that of a previously defined unit.

When a job running in iSTREAM mode submits another job, by default this other job inherits the mode. It is achieved by using `CPYENVVAR(*YES)` parameter of `SBMJOB` command. By default, iSTREAM creates a copy of `SBMJOB` command in its system library, changes the default value of `CPYENVVAR` parameter for this command to `*YES`, and places `ISTSSYS` library at the top of the system part of the library list making sure that `ISTREAM_UNIT` environment variable is always copied when a new job is submitted. If a job, while in iSTREAM mode, executes `SBMJOB` command explicitly setting `CPYENVVAR` parameter to `*NO`, the new job does not inherit iSTREAM mode. The latter job, however, may still have `ISTSSYS` library as the top library of the job library list.

If STRISTMOD commands executed from multiple unit jobs (jobs manipulating data in the unit libraries) define different iSTREAM mode parameters, iSTREAM behaviour may be unpredictable.

If a job is already executing in the iSTREAM mode and some of the mode parameters have to be changed, it is important to understand that other jobs

running in the iSTREAM mode for the same unit may not immediately be affected by the change due to iSTREAM parameter caching. In order to update the cached iSTREAM configuration values, STRISTMUD command can be executed in such jobs.

Warning: Caching is extensively used throughout iSTREAM, so any changes to the iSTREAM configuration or command enablement for different types of transformations performed in a job may not become immediately available to other jobs of the system without reclaiming iSTREAM activation groups CRCV and CRCS in those jobs.

ENDISTMOD command destroys the iSTREAM environment in the current job. iSTREAM configuration objects, however, remain in the iSTREAM system data library (ISTVQS) and, in certain cases, in the control library of the unit.

The use of STRISTMOD command is generally risky, since it can be used to change iSTREAM unit configurations. Once the unit configuration has been defined, STRISTMUD(Start iSTREAM mode with unit defaults) command could be used instead. This command enters the iSTREAM mode for the current job using previously defined unit parameters.

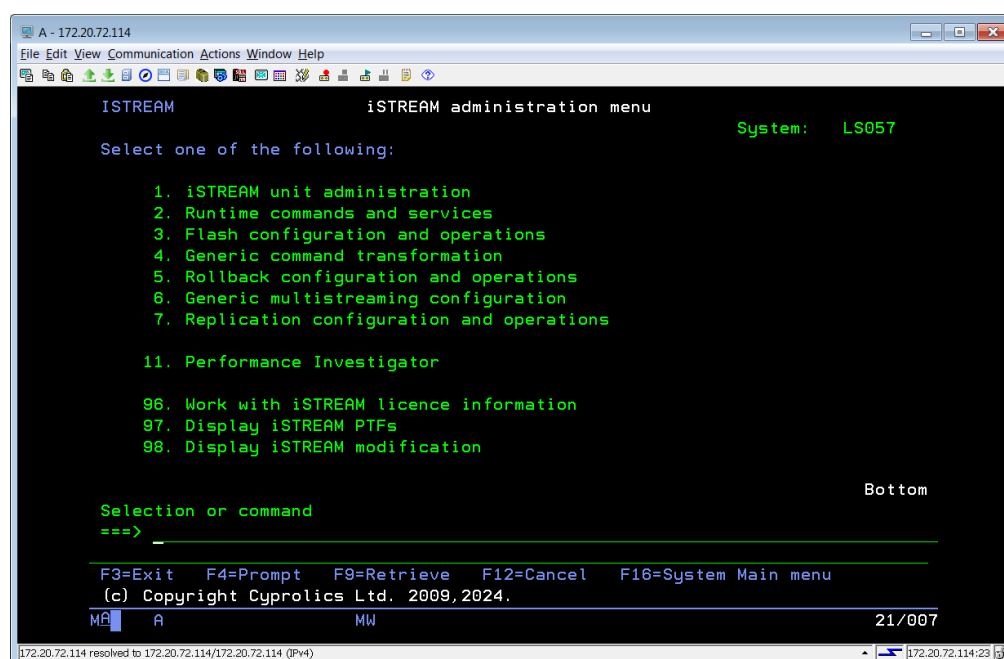
iSTREAM programs make extensive use of caching, so in order to consistently switch iSTREAM mode from one unit to another it is necessary to explicitly end the current mode (ENDISTMOD).

2.3 Common unit (@@@)

It may well be that changing jobs to make them execute STRISTMOD/STRISTMUD command in order to benefit from iSTREAM functionality is inconvenient. In such cases common unit (@@@) functionality can be used. Unit with the name @@@ is automatically created when iSTREAM licensed product is installed. By default, all functions in the unit are disabled, but they can be enabled just like for any other unit. The main difference between the common unit @@@ and any other unit is that if ISTSSYS library is in the library list and STRISTMOD command has not yet been executed in a job, the job is still considered to be in iSTREAM mode for unit @@@.

2.4 Introduction to interfaces, commands and menus

iSTREAM runtime comes with iSTREAM Unit Administration Menu. All iSTREAM system commands are available from this menu. iSTREAM main menu screen is shown in the picture below.



iSTREAM runtime includes the following user commands:

- STRISTMOD – to start iSTREAM mode
- STRISTMUD – to start iSTREAM mode with unit defaults
- ENDISTMOD – to end iSTREAM mode
- CHKCMDENA - to verify and attempt to fix (if a failure is detected) QIBM_QCA_CHG_COMMAND and QIBM_QCA_RTV_COMMAND exit point definitions for the command objects enabled for asynchronous execution, user-defined transformation and rollback.
- RTVENASRC - to retrieve the source required to enable CCT transformations (flash, user tfm and rollback) for the commands defined in ISTVQS/ISTVCMD file. The source is retrieved into member CMDSRC of the specified source file
- RSTUNTCFG - to restore an earlier saved unit configuration including the hot library, replication control library, transformation source file, control library objects (checkpoint file is not restored but recreated if it does not exist) and related command transformation enablement definitions. The command ends iSTREAM mode on exit

iSTREAM Command Transformer (CCT) Guide

- SAVUNTCFG - to back up the entire unit configuration including the hot library, replication control library, job descriptions and user exit programs created in the control library (the checkpoint file is not saved), command transformation source file, and the related command transformation enablement definitions
- RQSUNTDMP - to request a complete unit dump for remote analysis.
- CMPPRMLST - to compile a protected unit libraries from source and makes the list available as target for *PRMLST reference from LIBLIST parameter of STRISTMOD command.
- DLTUNTCFG - to delete iSTREAM unit configuration objects. Replication control library and hot library are removed, if exist. Unit control library is not removed, but iSTREAM exit programs, automatic rollback job descriptions, and the checkpoint file are removed from it, if found.

The above commands are included in the *BASE option of the iSTREAM product.

2.5 Proxi command objects and chain transformations

Generally, proxi command objects can be used to define command transformations provided by Options 1, 2 and 3 of iSTREAM. When proxi command objects are used for enabling transformations of this or that type the actual commands configured for transformations are use as the related target commands. For example, enabling QSYS/SAVRSTLIB for flash may cause the related SAVRSTLIB command residing in QSR library to be used by the registration process.

Nevertheless, proxi commands can be used both in iSTREAM configuration definitions and for the Enable and Disable operations.

Enabled commands are usually transformed in the following sequence:

- customer-defined transformations (Option 2)
- rollback transformations
- transformations for flash

Explicitly qualified commands, e.g. QSR/SAVRSTLIB cannot be transformed by iSTREAM. Qualifiers *LIBL and *SYSTEM are allowed.

2.6 iSTREAM APIs

2.6.1 "Submit concurrent batch jobs" framework

This framework is a by-product of the CCT runtime environment. Different iSTREAM 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, CRCVSPP and CRCVSPW. Both are included as procedures in ISTVCOR0 service program. CRCVSPP is used to submit named groups of multiple concurrent jobs, CRCVSPW – to wait for completion of all jobs belonging to a group.

CRCVSPP 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 synchronisation objects (parameter 6)
2. Input DEC(4,0) – 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(5000) – CL command used to initiate the job's execution stream
4. Input CHARACTER(10) – name of the job description to be used (must be available via the library list of the submitting job)
5. Input/Output CHARACTER(7) – effective identifier of the group to be created. It may be 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) – effective unique identifier of the group
2. Input DEC(4,0) – number of processes in the group
3. Input CHARACTER(5) – maximum time to wait for the processes to complete in minutes. The maximum value that could be specified is 32767, which is the same as a special value *MAX.
4. Output CHARACTER(1): response code ('1' if errors in one or more submitted processes have been ignored, and '0' otherwise)
5. Input CHARACTER(10) – name of the synchronization library

In principle, background jobs could be concurrently submitted to different groups having different effective identifiers. Only one CRCVSPW call can, however, be made at any one time, meaning that waiting for jobs from multiple groups using a single CRCSSPW call is not possible.

If all the jobs belonging to a group with a certain effective identifier need to be cancelled it can be done using CRCSCSJ API. This API has only one parameter, the effective identifier of the group.

CRCVCSJ parameters:

- Input CHARACTER(7) – effective unique identifier of the group

All three APIs are implemented as procedures of ISTVCOR0 service program residing in ISTSSYS library.

2.6.2 Other APIs

CRCVOPL API can be used to find an object in the job library list.

CRCVOPL parameters:

- Input CHARACTER(10) – object name
- Output CHARACTER(10) - the first library containing an object with the given name
- Input CHARACTER(10) - exclusion library list (libraries omitted from search)
- Input CHARACTER(10) - default library name returned if the object is not found
- Input CHARACTER(10) - object type

This API is implemented as an ILE program residing in ISTSSYS library.

Warning: The above APIs can only be used in a job executing in iSTREAM mode.

3. Flash Operations – Option 1

3.1 Introduction

This feature allows to automatically change the mode of execution for selected IBM i CL commands submitting them to the background. Additionally, commands of the SAV group are transformed to use save-while-active (or flash) functionality.

For the transformation to take place target CL commands must be defined to iSTREAM using ENAFLSEXE command, and flash execution – enabled for the library unit (parameter ASYEXE of STRISTMOD command).

Save-while-active parameters used are SAVACT(*LIB), SAVACT(*SYNC) or SAVACT(*SYNCLIB), depending on the command used, and SAVACTWAIT(///// mmmmm nnnnn). By default, the value ///// is 00900. It can, however, be changed by using CHGDFTIST command.

The actual system commands that iSTREAM attempts to submit to the background have to be defined using EANFLSEXE iSTREAM command. *ISTDFT generic name can be used to enable background execution of the following system commands (command objects found using *LIBL):

- SAVOBJ
- SAVLIB
- SAVRSTCHG*
- SAVRSTOBJ*
- SAVRSTLIB*
- SAVLIBBRM
- SAVOBJBRM
- CHKTAP
- DSPTAP
- RST
- RSTLIB
- RSTOBJ

The above commands are by default enabled for flash execution when Option 1 is installed and there are no exit point number conflicts (see section 3.8 for detail).

Commands with no support for the save-while-active mode are simply submitted for execution to the background.

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

Commands marked with the star symbol ("*") are by default enabled for flash transformation but can only be transformed at the second step of a two-step or a multi-step transformation procedure (see section 6 for details). This is due to a restriction of the IBM command transformation exit point QIBM_QCA_CHG_COMMAND related to commands having parameters with either DSPINPUT(*NO) or DSPINPUT(*PROMPT) property.

In certain cases if STG(*FREE) parameter is used, QTEMP references found, or save-while-active mode used in the original command strings, commands identified for background execution are instead executed inline. Inline here means that commands are executed in the job that issued the original command request. DSPTAP command is executed inline if the value of OUTPUT parameter is '*'.

Flash operations are available in two modes. If ASYEXE parameter is set to '*YES' on STRISTMOD command, all flash requests are executed serially. 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 ASYEXE parameter is set to '*INDEP' by STRISTMOD command, the logical flash server does not execute requests arriving from the request queue inline; instead, it submits those requests 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 flash server does not in this case serialise or synchronise requests in any way. In this mode synchronisation must be guaranteed by application developers. The only synchronisation tool provided by CCT is WAITFLSRQS command. This command, when issued by an application program, instructs the flash server for the unit to accumulate new requests without processing them until all previous requests, including those submitted as independent jobs, have been completed. Additionally, the command puts the issuing job in the wait state until that time. CHKTAP and DSPTAP command processors issue WAITFLSRQS command with SCOPE(*UNIT) parameter under the covers when executed inline or by independent flash processes (ASYEXE(*INDEP)).

3.2 Configuration and function

In order for the flash operations facility to be activated, option 1 must be installed and ASYEXE parameter of the last STRISTMOD command executed for the iSTREAM unit mode the job is running in must have the value *YES or *INDEP. Target commands must be enabled for flash execution using ENAFLSEXE command. The flash operations feature makes use of subsystem ISTREAM@@@ and two job queues, one for synchronised (dependent) and the other for independent requests. The work management configuration is created automatically.

If required, ISTREAMxxx subsystem can be created by cloning and editing ISTREAM@@@ configuration. It is then used for running requests submitted on behalf of jobs running in the xxx unit. Job queues for the ISTREAMxxx subsystem must be created under the names of ISTREAMxx1 and ISTREAMxx2, respectively. By default, job descriptions with the same names will be used. Routing entry data used is retrieved from the job descriptions. In order to create a template flash subsystem for a unit the following program call be used:

```
CALL ISTSSYS/CRCVCFS PARM(&UNIT)
```

ISTREAM Command Transformer (CCT) Guide

In the default configuration only 5 jobs submitted to ISTREAM@@1 queue can be executed concurrently. This means that only 5 units can process their synchronised flash requests at the same time.

Each supported command, except for those using save-while-active mode, is intercepted and put on work queues in ISTVQS library for flash execution. For synchronisation the following message queues are created in ISTVQS library:

- SS1 - global system queue used by WAITSYRQS SCOPE(*SYSTEM) wait requests
- SS2xxx - unit-level queue used by WAITSYRQS SCOPE(*UNIT) wait requests
- SS3xxxxxx - job-level queues used by WAITSYRQS SCOPE(*JOB) and SCOPE(*KEY) wait requests
- RQMxxxxxx - job-level save-while-active checkpoint queues

SAVRQSxxx files store information about requests currently executing in related units.

When a request is intercepted its syntax is checked and a job is submitted to the appropriate job queue. Control, however, does not return to the user program until the submitted job starts and the save-while-active checkpoints have been successfully taken. If the system cannot take the checkpoints, an error message is returned to the user program.

Serialised requests (ASYEXE(*YES)) are executed one at a time for each unit.

If ASYEXE parameter of the STRISTMOD command is set to *INDEP, the requests are submitted for execution to another job queue configured to run multiple jobs. These requests are also called independent flash requests.

For non-SAV commands enabled for flash execution processing the logic is exactly the same as above, but save-while-active checkpoints are not taken.

The type of the request is determined at the time of its creation and does not change when ASYEXE mode of the related unit changes. Even if flash processing is disabled for the unit, already submitted requests will continue to be processed until they either end or are cancelled.

Flash command execution processors (jobs) are usually submitted with predefined job description names. If, however, prior to the execution of a command enabled for flash execution the name of a job description residing in ISTVQS library is stored in PARMJOB data area in QTEMP, this job description is used.

In order to use flash copy functionality involving such system commands as SAVLIB, SAVOBJ, RSTLIB, RSTOBJ, SAVRSTLIB, SAVRSTOBJ, etc. the user has to be authorised to the required command objects and Object Connect programs in library QSR.

Service jobs performing background flash-related tasks can be assigned to a workload capping group (WLCASY parameter of STRISTMOD command). The value of the parameter is only used when the group exists.

3.3 Remote Flash option

3.3.1 Configuration and Operation

In addition to ASYEXE, STRISTMOD command has ASYDDM parameter that could be used to specify a name of a DDM file. If the value of ASYEXE parameter is *YES or *INDEP, and if a DDM file is specified as the value of ASYDDM parameter, this DDM file must point to a file in a remote partition where a copy of the iSTREAM LP is installed and licensed. A unit with the same name must be defined in this remote partition. It is not required but strongly recommended that the remote unit be defined exactly as the local one.

Remote flash functionality allows to offload flash operations (usually, backup commands) to the remote partition, where copies of the production libraries are maintained, using some data replication application. In the case of Precisely MIMIX, additional handshaking can be defined in order to synchronise the copies of the database.

Remote Flash makes use of the same queues, subsystem and job descriptions as the Flash in the local partition. The difference is, instead of executing the target command inline in a job running in the same partition as the request originator job, the command is submitted by the same service job to the remote partition using SBMRMTCMD system command. Therefore, if, for example, DSPFLSEXE command were used to display the status of a background request, selecting option 5 would take the user to the job submitting the remote command. The command itself will be executed in a job started from the communications entry of the relevant subsystem in the remote partition. It is important to point out that save-while-active checkpoints taken by Remote Flash are taken in the remote partition and relate to the remote database.

If a unit is used for Remote Flash on the remote system, it is strongly recommended that it be used just for this purpose; mixing local processing and remote requests in the same unit has a potential of creating confusion.

If Option 3 is installed on the local system, a Rollback checkpoint is taken each time a Remote Flash request is submitted for execution. If MIMIX handshaking is configured as the default CRTCKP option, synchronous checkpoint is taken for both PROD and DR systems.

DSPFLSRQS and WAITFLSRQS commands can be used, exactly as in the case of the Local Flash, in the local partition. The former, however, only displays requests as they are known to the local system and service jobs executing SBMRMTCMD command. For the actual details of the flash background processing it may be required to analyse the processes on the remote system.

Workload capping group configuration in the case of remote flash only relates to the service jobs on the source system.

3.3.2 *Network configuration*

The DDM file required for configuring Remote Flash must be created according to the principles defined in the Distributed Data Management (DDM) sections of the IBM i Communications Management manual.

The DDM file used with the Remote Flash can be configured to use either *SNA or *IP protocols. The use of *SNA provides more flexibility in terms of security and, therefore, is recommended. If *SNA protocol is used, the related APPC link between the two partitions has to be created, including line, device, and controller definitions. Also, a communications entry for the DDM file mode and location has to be added to the subsystem the APPC target device is allocated to. The APPC configuration used must have ISTREAMR mode defined and started for the source and target devices.

3.3.3 *Security settings*

The profile used to execute the remote SBMRMTCMD requests must have access to all the command objects in the target system used as part of Remote Flash requests. It is best to select a profile (using the related communications entry) with the lowest possible authority. If additional authorities are required on the remote system to execute the background process, the exit program ISTFUX01 can be created in any of the libraries accessible from the remote server job via *LIBL or ISTSSYS library. This program is executed before the actual flash request is submitted by the server job. ISTFUX01 program has to be called with the following parameters:

- UNIT (CHAR 3)
- Response code (CHAR 1) - '0' (success) or '1' (failure). Any other code would be treated as an internal error.

The server job executing the actual flash request runs in the iSTREAM mode for the unit.

3.4 Commands and menus

The option includes the following user commands:

- ENAFLSEXE – to enable a CL command for flash execution
- DISFLSEXE – to disable flash execution for a CL command
- LSTFLSEXE - to output the list of commands currently enabled for flash transformation
- LSTFLSJOBA command is an API that can be used to determine whether any of the earlier submitted flash server processes require attention. Optionally, it also outputs the list of such processes to a file. The API provides a single-character response code. The value '0' means that no background flash processes require attention. The value '1' stands for at least one background flash process requiring manual interference.
- DSPFLSRQS - to display the currently executing requests
- DSPFLSRQD - to display the currently executing request details
- WAITFLSRQS - to wait for the currently pending flash request of the specified scope. The command has SCOPE parameter the value of which can be set to *JOB, *UNIT or *SYSTEM. SCOPE(*SYSTEM) causes the current job to wait for all outstanding flash requests in all units. All new requests are suspended and related jobs put in the wait state until the WAITFLSRQS wait is through. SCOPE(*UNIT) works in a similar way but only for requests submitted in the specified unit. SCOPE(*JOB) means waiting only for the asynchronous processes submitted from the current job in the specified unit. In that case WAITFLSRQS command does not take into account requests created by other jobs.

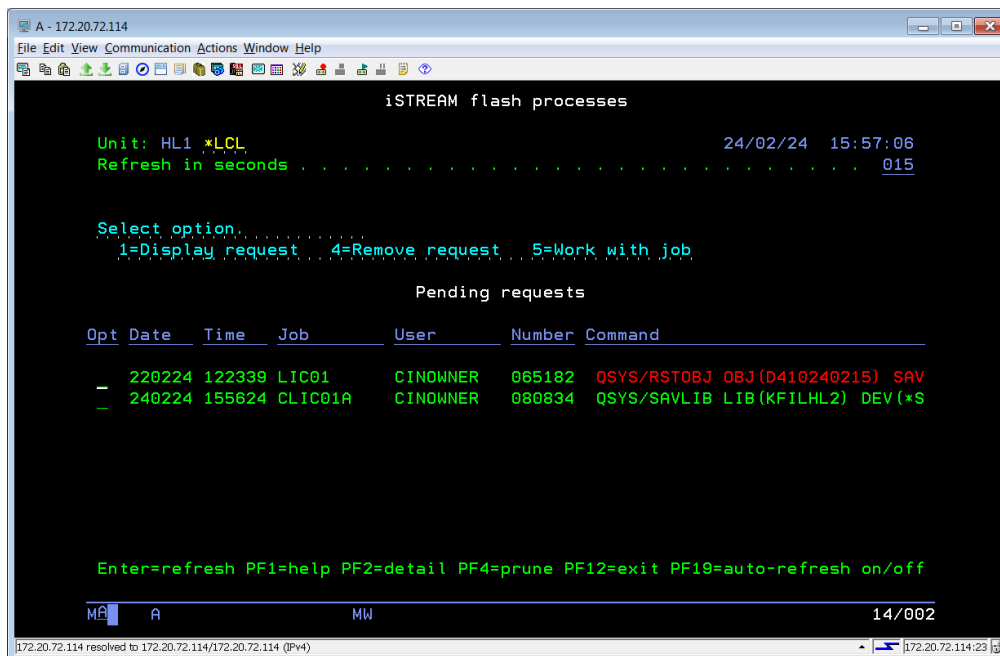
Each time a new flash request is submitted for asynchronous execution, a unique key of the request is returned to the invoking process in CRCVKEY data area in library QTEMP. This data area is defined as CHARACTER(38), and the key value includes all the job identifiers, as well as the date and the time of the request. The value of the request key can then be used when SCOPE(*KEY) is specified as the scope definition parameter of WAITFLSRQS command. In that case, WAITFLSRQS waits for the request with the given key in the given unit to end and then returns control to the calling process. If no such request exists, control is immediately returned to the calling process.

WAITFLSRQS has the timeout parameter. Waits exceeding the specified timeout duration end in error.

CRCVEX9 utility can be used to globally reset all pending flash(asynchronous) requests in a partition. The actual jobs that may be executing the requests are not affected. It is recommended to only use this utility in quiet periods with no other iSTREAM activity in the partition.

iSTREAM Command Transformer (CCT) Guide

DSPFLSRQS command displays all outstanding requests. The monitor screen it displays may look like this:



```

A - 172.20.72.114
File Edit View Communication Actions Window Help

iSTREAM flash processes

Unit: HL1 *LCL 24/02/24 15:57:06
Refresh in seconds . . . . . 015

Select option.
1=Display request 4=Remove request 5=Work with job

Pending requests

Opt Date Time Job User Number Command
- 220224 122339 LIC01 CINOWNER 065182 QSYS/RSTOBJ OBJ(D410240215) SAV
- 240224 155624 CLIC01A CINOWNER 080834 QSYS/SAVLIB LIB(KFILHL2) DEV(*S

Enter=refresh PF1=help PF2=detail PF4=prune PF12=exit PF19=auto-refresh on/off

MA A MW 14/002
172.20.72.114 resolved to 172.20.72.114/172.20.72.114 (IPv4) 172.20.72.114:23

```

It supports automatic refresh and options for working with outstanding requests and server jobs.

Additional information about independent requests can be obtained by pressing F2 from the main flash monitor status display.

When option 7 of iSTREAM is installed, STRFS command is enabled for flash by the installation procedure. The command object for STRFS resides in ISTSSYS library. In order to prevent inadvertent removal of this command object from the list of commands enabled for rollback, ENAFLSEXE, DISFLSEXE and LSTFLSEXE functions have two modes of operation, *USER and *SERVICE. The latter mode is used by iSTREAM programs, the former is the default mode. Users can override this mode by specifying OPTION(*SERVICE) parameter for the above iSTREAM commands.

3.5 CPYLIBF command

CPYLIBF command included in the CCT distribution represents a possible workaround for the IBM synchronisation problem in SAVRSTxxx commands where entries in the system directory are not updated immediately. This delayed update may cause issues when, for example, the target SAVRSTxxx library is used in scenarios involving prior deletion of the same library.

1

CPYLIBF implements SAVRSTLIB functionality by saving the source library to a temporary save file and then restoring it to the target library. In order to use CPYLIBF instead of SAVRSTLIB, CPYLIBF command should be included in target sections of iSTREAM command transformation definitions instead of SAVRSTLIB. Alternatively, CPYLIBF can be used independently of iSTREAM as a replacement for SAVRSTLIB. CPYLIBF only supports intra-system SAVRSTLIB operations and has two parameters, FROM and TO library names. *SQLPKG objects are omitted by SAVLIB invoked from CPYLIBF processing program. Error messages generated by SAVLIB and RSTLIB commands are not processed by CPYLIBF processing program.

3.6 SBMICMD command

SBMICMD command is a simple interface to QCMDEXC command invocation API. If SBMICMD is enabled for flash execution, it can then be used to submit generic jobs to the iSTREAM flash subsystem benefiting from the iSTREAM flash execution synchronisation and monitoring features described above.

3.7 SAVACT option override

A copy of CRCVSOP file residing in ISTSSYS library and containing SAVACT options for each system command supported by Flash Feature can be optionally created in ISTVQS library under the name CRCVSOP_{unt}, where *unt* is the name of the library unit. Thus, SAVACT options used by iSTREAM by default can be overridden.

If the customised file does not exist or the SAVACT value for a command in this file is blank, the related value is retrieved from CRCVSOP file in ISTSSYS library.

3.8 Exits defined

Flash Operations facility optionally makes use of QIBM_QCA_CHG_COMMAND exit points with number starting from 1882110101 (maximum 100 commands can be enabled for transformation). Flash Operations (option 1), Command Transformation (option 2) and Command Rollback (option 3) share the command object space and exit points. If any other product makes use of the same entries, Option 1 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 7S77STR installation exit program (by default, the number used is 1882110001).

3.9 Service trace

If CRCOPT1 message queue is available via *LIBL at the time of the flash copy request processing, the target CL command submitted for background execution is copied to this message queue.

4. iSTREAM Command Transformation – Option 2

4.1 Introduction

This option enables definition of configurable command transformations where IBM i CL commands can be transformed or replaced at runtime.

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.

4.2 Configuration and function

4.2.1 Functional description

Transformation details for CL commands executed when the process is running in iSTREAM mode can be defined in the transformation source file. This file must be created with the record length of 112 bytes. In order to locate this file iSTREAM uses the following search order:

- if CRCVCMD unt data area, where unt is the name of the unit, can be found in ISTVQS 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, iSTREAM checks CRCVCMD unt file in ISTVQS 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 ISTVQS library. In case this file does not exist command transformation facility for the unit is disabled

It is strongly recommended, however, to create transformation files in user libraries thus avoiding complications related to mixing user source and iSTREAM configuration objects in the same library.

To transform a command executing in iSTREAM mode this command with all its runtime parameters must be defined in the source section of a CRCVCMD unt definition member. If, for instance, at runtime

```
SAVLIB LIB(KFILPRO) DEV(TAP01) TGTRLS(V7R4M0)
```

command is issued and it is desirable to modify it before execution, a new transformation definition must be created with the source section containing the command string followed by the separator line (see examples for detail).

Single '+' and '-' signs at the end of each line in the template member are removed and multiple blanks transformed into single blanks at runtime before the command issued by the application is compared with the source string. Blanks, however, are not removed from literals.

iSTREAM Command Transformer (CCT) Guide

If literals are used in the comparison template, they cannot span multiple lines.

The command transformation member must have the name prefix of the command being transformed. The target section of the definition follows the separator line (`/**/`), has the same structure as the source section and can define any CL command. If option 1 of iSTREAM is also installed and activated, commands after transformation can be submitted for asynchronous (flash) execution. For that, the target command of the transformation definition must be enabled for flash execution (see section 3 for detail).

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` is replaced by the command transformation processor with the name of the unit, and `@BK` – with the contents of `*CHAR(3) CRCVBKPID` data area, if it is found via `*LIBL`, `@PRDAT` – with the current processing date value in the same format as `RTVSYSVAL SYSVAL(QDATE)` would return, and `@EP` – with the Finastra Equation End-of-Day phase identifier.

`@CMD` variable can be used in the target section of the definition to include the entire original command into the target command string.

Lines of the definition starting from the right slash, star and blank characters,

```
'/* '
```

are considered comments and ignored by the transformation processor.

Another separator line (`/**/`) must follow the target section of the definition.

Members of `CRCVCMD` source file are processed in the alphabetical order, and when a match is found the search is terminated.

It is possible to condition transformations 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 a CL-compatible logical expression containing substitution variables `&UN`, `&BK`, `&EP` and `&PRDAT`. The first character of the variables that can be used to define conditional processing is ampersand (`&`). Conditional transformation, however, does not have to be specified using the member description field. It can also be achieved by adding the following line to the actual transformation source:

```
/*IF:<condition> */
```

The closing bracket `*/` is optional. For example,

```
/*IF: (&BK='PT1') */
SAVLIB LIB(KFIL@UN) DEV(TAP01) +
  TGTRLS(V7R____)
/**/
SAVLIBBRM LIB(KFIL@UN) DEV(*NONE) MEDPCY(*NONE) +
  TGTRLS(V7R4M0) DTACPR(*YES) EXPDATE(*PERM) +
  MOVPCY(*NONE) MEDCLS(SAVSYS) LOC(*ANY) +
```

iSTREAM Command Transformer (CCT) Guide

```

    SAVE(*YES) SAVFASP(1) SAVFEXP(*NONE)    +
    MAXSTG(1) VOLSEC(*NO) MINVOL(1) MARKDUP(*NO)
  /**/

```

The condition cannot be longer than 50 characters; the remaining characters are ignored. If a condition is defined in both the member descriptor field and the text of the transformation member, the former overrides the latter.

In order to enable the conditions defined, they must be compiled using CMPCMDTFM command. If any conditions change, CMPCMDTFM compilation has to be used again in order to refresh the earlier compiled condition implementation programs. CMPCMDTFM command may be executed manually, but when the command transformation mechanism is triggered (command enabled for transformation is executed in a unit where transformation has been enabled), iSTREAM always verifies the compiled version of the conditions by comparing the creation date of the related program with the change date of the transformation definition file for the unit. If the condition verification program does not exist or its creation date is earlier than the change date of the transformation definition file, CMPCMDTFM command for the unit is executed implicitly. Since CMPCMDTFM may fail due to syntax errors in definitions, it is always recommended to execute CMPCMDTFM manually after each update of the transformation definition file.

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

The role of CMPCMDTFM command is not limited to the compilation of conditional transformation. Actually, this command creates a file with the name of CRCVCFAunt in ISTVQS library. This file is used to store the entire transformation configuration for the unit. The source transformation definitions are not changed and, should any of them be modified, the CMPCMDTFM process, unless explicitly requested, is executed under the covers at the time of the next transformation. The compiled transformation definition object (user space cache) is used instead of the source file for better performance at runtime.

Chains of transformations, e.g. CPYF to SAVRSTOBJ and SAVRSTOBJ to FTP, are not supported. If a chain of transformations is effectively defined, runtime outcomes are generally unpredictable.

4.2.2 Command Transformation Example

To substitute

```
SAVLIB LIB(KFILPRO) DEV(TAP01) TGTRLS(V7R3M0)
```

and

```
SAVLIB LIB(KFILPRE) DEV(TAP01) TGTRLS(V7R3M0)
```

commands, PRO and PRE being the names of iSTREAM library units, with

```

SAVLIBBRM LIB(KFILPRO) DEV(*NONE) MEDPCY(*NONE) TGTRLS(V7R4M0)
DTACPR(*YES) EXPDATE(*PERM) MOVPCY(*NONE) MEDCLS(SAVSYS)
LOC(*ANY) SAVF(*YES) SAVFASP(1) SAVFEXP(*NONE)

```


iSTREAM Command Transformer (CCT) Guide

```
MAXSTG(1) VOLSEC(*NO) MINVOL(1) MARKDUP(*NO)
```

and

```
SAVLIBBRM LIB(KFILPRE) DEV(*NONE) MEDPCY(*NONE) TGTRLS(V7R4M0)
DTACPR(*YES) EXPDATE(*PERM) MOVPCY(*NONE) MEDCLS(SAVSYS)
LOC(*ANY) SAVF(*YES) SAVFASP(1) SAVFEXP(*NONE)
MAXSTG(1) VOLSEC(*NO) MINVOL(1) MARKDUP(*NO)
```

commands respectively,

SAVLIB1 member containing the following definition can be added to each of the CRCVCMDunt files in ISTVQS as follows:

```
SAVLIB LIB(KFIL@UN) DEV(TAP01) +
  TGTRLS(V7R____)
/**/
SAVLIBBRM LIB(KFIL@UN) DEV(*NONE) MEDPCY(*NONE) +
  TGTRLS(V7R4M0) 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 50 lines excluding the separator line.

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

```
CHKTAP DEV(TAP01)
/**/
/**/
```

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

In order to be processed asynchronously (in flash mode) commands must be supported by Option 1 and issued with no qualifiers. For example, neither QSYS/SAVOBJ, nor CALL PGM(ABCD) commands can be sent to the flash server for execution, even if the match is found and they are selected as target commands by the transformation mechanism. However, if ABCD program contains other CL 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 executed. If CMDTFM parameter value of STRISTMOD command is set to *SOURCE and certain commands are enabled for configurable transformation, command strings for all such commands executed by the application code while in iSTREAM mode are stored as messages in CRCVCMDunt message queue in ISTVQS library. Information in these messages can then be used to define command transformations. Since some of the command parameters are modified by iSTREAM CCT 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.

4.2.3 Parsed substitution variables

In addition to the above-described simple substitution variables, parsed substitution is also supported. There are ten variables of this type defined: @P0-@P9. Each of them has CHARVAR(50) type. Values are assigned to these variables when the pattern of an issued CL command matches the pattern of the command transformation source string defined in CRCVCMD*unt* file. These variables can therefore be mapped to certain substrings of the CL command string.

The transformation definition from 4.2.2 can be modified as follows:

```
SAVLIB LIB(@P1) DEV(TAP01) +
  TGTRLS (V7R____)
/**/
SAVLIBBRM LIB(@P1) DEV(*NONE) MEDPCY(*NONE) +
  TGTRLS (V7R4M0) 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 (V7R4M0)
```

command is then issued, iSTREAM 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 the parsing operation, assign the substring starting at the position of the parsed variable in the source definition string and ending before the next matched pattern is detected. In this example, @P1 will be assigned the value "ABCD". This library will then be backed up using BRMS.

The same parsed variable can be used several times in the same definition. If, however, different values are assigned to different instances of the same variable as a result of pattern mapping, "no match" condition is signaled. The maximum number of parsed variables in a single pattern is 10.

4.2.4 Partial matching

In some cases it may be difficult to determine the exact format of the source CL command that needs to be transformed. Partial (fuzzy) source command matching feature could then be used. An example of such transformation definition is given below.

```
/*PARTIAL*/
SAVLIB LIB(@P1) DTACPR(@P2)
/**/
SAVLIBBRM LIB(@P1) DEV(*NONE) MEDPCY(*NONE) +
  TGTRLS (V7R4M0) DTACPR(*YES) EXPDATE(*PERM) +
  MOVPCY(*NONE) MEDCLS(SAVSYS) LOC(*ANY) +
  SAVF(*YES) SAVFASP(1) SAVFEXP(*NONE) +
  MAXSTG(1) VOLSEC(*NO) MINVOL(1) MARKDUP(*NO)
/**/
```

Specification

```
/*PARTIAL*/
```

entered in the first line means that iSTREAM will match any SAVLIB command containing LIB and DTACPR parameters to this definition and perform the requested transformation. The order of the source command parameters is ignored when matching is performed.

Parsed variable and wildcard use in definitions of the PARTIAL type is limited to parameter values or their parts. Parameter names and related brackets must be entered in the matching template exactly as they are defined in the actual CL commands.

Control specification

`/*EXACT*/`

if entered in the first line, stands for the exact matching. If the type of matching is not specified, exact matching is assumed.

4.2.5 CKSTACK specification

The following specification can appear anywhere before the source command template in the transformation definition member:

`/*CKSTACK=PGMNAME` `*/`

or

`/*CKSTACK<>PGMNAME` `*/`

The program name is taken from the 10 positions of the record following the '=' or '<>' sign. The rest of the line is ignored.

The transformation definition with such specification is only processed if a program with the name PGMNAME is present (equal condition) or not present (not equal condition) in the calling stack of the current process. No quotes or other special characters are allowed inside the specification.

If either

`/*EXACT*/`

or

`/*PARTIAL*/`

specification is also used, it should precede CKSTACK specification which, in turn, should be included in the definition member before the text of the source command template.

4.2.6 Control specification sequence

Each transformation definition can contain the following sections:

```
<Section 0> (control specifications)
<Section 1> (source template)
/**/ (separator - no blanks)
<Section 2> (target template)
/**/ (separator - no blanks)
```

Section 0 is optional and can contain specifications of these types:

```
Matching type
CKSTACK
Conditional specification (IF:)
```

Comment lines can be included anywhere in Section 0.

The first line that does not start from the forward slash-asterisk ('/*') sequence is interpreted as the first line of section 1.

4.2.7 Prefix "++"

Sometimes, the command transformation required involves adding a parameter to the source command, rather than changing it into a different command. That's where the "++" can be used. For example, the following definition adds the PRECHK parameter to all SAVLIB commands executed in the iStream mode for a unit.

```
/*PARTIAL*/
SAVLIB LIB(@P1)
/**/
/*++*/ PRECHK(*YES)
/**/
```

In the above case the target command section specifies the string added to the command being transformed, rather than a target replacement command.

Multiple prefixes can be used in the same target section. For example, the above transformation with the additional singleton mode applied (See iSTREAM Generic Multistreaming Toolkit manual for details) may look as follows:

```
/*PARTIAL*/
SAVLIB LIB(@P1)
/**/
/*SL*/
/*++*/
PRECHK(*YES)
/**/
```

The target command in the previous example may be completely omitted, e.g.

```
/*PARTIAL*/
SAVLIB LIB(@P1)
/**/
/*SL*/
```

```
/*+*/  
/**/
```

causing the source command to be executed as a singleton.

4.2.8 CPYF multistreaming transformation

iSTREAM supports multistreaming of CPYF commands where source files are physical files. For that, iSTREAM Generic Multistreaming feature must be used in combination with the user-defined transformation feature of CCT. Since iSTREAM Generic Multistreaming feature can only be used to multistream programs, a program executing the given CPYF command must be created. In the following case iSTREAM Command Transformer is used to configure the transformation of the source CPYF command to a program call. It can be done with a definition similar to the one below.

```
/*PARTIAL*/  
CPYF FROMFILE(@P1/FILE1) TOFILE (@P2/FILE1) MBROPT(*ADD)  
/**/  
LIBRARY/@MS-CPYF  
/**/
```

The above definition means that copying file FILE1 from one library to another is enabled for multistreaming. @MS-CPYF is a special instruction to iSTREAM to create the required objects (command and program) for enabling multistreaming. LIBRARY is the name of the library where those objects are to be created. If the GUI front end of iSTREAM Access for MS Windows is used for entering such a definition, service objects will be created automatically when the definition is saved. Users of the 5250 interface should execute CRTTFMMS command to make sure that those objects are created. Multistreaming should then be defined for the program with the same name as the definition member of the transformation source file. This program is to be invoked via a command with the same name, both residing in the library with the name specified on the @MS-CPYF line – LIBRARY.

The program created by iSTREAM has two parameters (*CHAR(5000) and *DEC(15 5)). Only *RRN breakdown is supported (see iSTREAM Generic Multistreaming Toolkit manual for detail). The split has to be compiled with SDROP(*YES) parameter. The CPYF command having exactly the same format as the original command is executed by each of the streams. In a stream, however, CPYF is used to copy only the records from the related range. If the original command contained a range definition (FROMRCD, FROMKEY, etc.), multistreaming is not performed and the command is executed in the single-stream mode.

If the original command contains MBROPT(*REPLACE) parameter it is replaced by MBROPT(*ADD). To clear the file in such cases EXIT1 program defined by DFNSPTPRM command can be used (see iSTREAM Generic Multistreaming Toolkit manual for more detail).

4.2.9 Transformation enablement

Transformation of any CL command (e.g. DSPLIB) must be explicitly enabled for each such command. In order to do this, ENACMDTFM command must be executed for the related command object. To enable the transformation mechanism QIBM_QCA_CHG_COMMAND exit point is registered for the command object. Only 100 commands can be enabled for transformation at any point in time. The entries with numbers from the 1882110101-1882110200 range are used by default. CRCSEID data area can be used to redefine the base of 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.

DISCMDTFM disables command transformation for the given command object.

When Option 2 is deleted all previously defined QIBM_QCA_CHG_COMMAND exit points are processed and either removed from the system or reregistered (for example, if flash execution for the related commands has also been defined and option 1 of the product is still installed). It is advisable to create a CL program containing ENACMDTFM commands for the required system command objects and execute it each time the current release of Option 2 is deleted and a new release installed.

4.3 Substitution variable name modifier

The special character used in the names of substitution variables in the body of transformation definitions (by default, '@') can be redefined. For example, the ampersand symbol can be used instead. In order to do this, CHGISTDFT command could be used to change the value of PVCH parameter. In the case of the ampersand ('&'), the names of the variables recognised by the transformation processor would be &UN, &BK, &EP, &PRDAT, &P0-&P9.

4.4 Commands and menus

iSTREAM command transformation line commands are:

ENACMDTFM – enables configurable command transformation

Warning: Transformation for some of the IBM i system commands cannot be enabled. The list of recognised exceptions in the current release includes the following commands: CLRMSGQ, DLTMSGQ, CHGJRN, RTVNETA, DLTCMD, OPNDBF, RTVSYSVAL, RTVDTAARA, SNDPGMMSG, RTVOBJD, CHKOBJ, HLDJOB, DLCOBJ, DSPSPLF, CHKPRDOPT, STRQMQR, RCVF, RTVJOBA, RTVNETA, DSPLOCKEY, STRCMTCTL, CRTDUPOBJ, DLYJOB, DLCOBJ, DMPCLPGM, RCLRSC, RCLACTGRP, ENDCMTCT. These commands cannot be enabled for rollback, user-defined transformation or asynchronous execution. If ENACMDTFM command is used to enable transformation for any of the “forbidden” commands, it will return an error message.

Transformation generally cannot be defined for command objects in ISTSSYS library, as it may cause iSTREAM functions to fail. Exceptions to this rule include a couple of rollback-specific service commands and CPYLIBF.

- DISCMDTFM – disables a previously defined command transformation
- CMPCMDTFM – compiles command transformation definitions. The command is executed implicitly when iSTREAM command processor exit detects a change in transformation definitions for the unit. Nevertheless, it is recommended that this command is executed manually in order to avoid transformation errors at runtime
- LSTCMDTFM – lists system commands with 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
- CRTTFMMS – creates CPYF service objects for multistreamed execution.

CRTTFMFIL command both creates the new source file and links it to the unit using the pointer data area CRCVCMDxxx in ISTVQS. 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 off using CMDTFM parameter of STRISTMOD command. This parameter takes effect immediately.

All iSTREAM Option 2 system commands are available either via "CMD Transformation" menu group of the iSTREAM Access for MS Windows GUI or 5250 CMDTF menu.

4.5 Exits defined

Command Transformation facility optionally makes use of QIBM_QCA_CHG_COMMAND exit points with number starting from 1882110101 (maximum 100 commands can be enabled for all types of transformation). Flash Operations (option 1), Command Transformation (option 2) and Command Roll-back (option 3) share the command object space and exit points. 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. The 10-character data area must contain the first number of the range to be used by 7S77STR installation exit program (by default, the number used is 1882110001).

4.6 Audit trail

If the system audit control variable (QAUDCTL) defines object-level audit (*OBJAUD), successful command transformations are recorded in the system audit journal. The type of journal entries is '4F'.

If CRCVMAPX1 program is added to the control library of the unit, it is called at the end of each transformation operation with the following parameters:

- Command name (CHARACTER 10)
- Command name length (DECIMAL 15,5)
- Unit name (CHARACTER 3)
- Original command (CHARACTER 5000)
- Original command length (DECIMAL 5,0)
- Transformed command (CHARACTER 5000)
- Transformed command length (DECIMAL 5,0)

This user exit program should not attempt to redefine any of the parameter values.

The size of the character strings notwithstanding, both command strings to be transformed and output command strings cannot exceed 5000 characters.

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 troubleshoot errors in configured transformations, Option 2 service trace facility can be used. Service trace is activated automatically if the job is placed in iSTREAM mode with command transformation enabled and if a message queue

iSTREAM Command Transformer (CCT) Guide

with the name CRCOPT2 is available via *LIBL. Transformation trace messages are sent to this message queue and can then be displayed using DSPMSG system command.

Diagnostic messages have 4-character identifiers uniquely identifying the reason the given substitution definition could not be used.

| Code | Description |
|---------------------|---|
| | |
| *INIT | Trace initialisation including values of available substitution variables. |
| *POS1 | Template part of the definition is too long (longer than 5000 characters). |
| *SYNT1 | Missing section separator /**/ after the command template section. |
| *COND | Member skipped due to the condition mismatch |
| *CKSTACK | Member skipped due to the unsatisfied CKSTACK condition (= or <>) |
| *POS2 | Target part of the definition is too long (longer than 5000 characters). |
| *SYNT2 | Missing section separator /**/ after the target command section. |
| *SYNT3 | Template part of the definition is empty (has 0 length). |
| *SYNT4 | Either template or target section is longer than 5000 characters after compression (removal of blanks between parameters). |
| *LENGTH | Command length is not equal to template length after variable substitutions. |
| *NMATCH, *DIFVAR | Command string could not be mapped to the source section of the template. |
| *VAR | This is a trace record including the value the parses attempted to assign to one of the parsed variables in the definition. |
| *FINAL | Final message of substitution member processing. |

5. Rollback for Library Units – Option 3

5.1 Introduction. Object protection

5.1.1 Object "protection" components

iSTREAM CCT can be used for "protecting" objects in unit libraries from the effects of unrecoverable commands, such as CLRPFM, CPYF OPTION(*REPLACE), etc. that cannot be rolled back using system journals. This feature helps mitigate the risks related to flash execution of long-running backup commands. This protection is configured for a unit using STRISTMOD command.

The way this protection works is, CLRPFM and similar CL commands are dynamically transformed into invocations of ILE processes natively deleting all records from the target file members.

CL command transformation of "CLRPFM" type has two modes of operation controlled by NREC parameter of STRISTMOD command, standard and parallel. Standard mode is used when the value of NREC parameter is *NONE or the number of records in the target file is smaller than the value of NREC parameter. In other cases parallel "CLRPFM" mode is used. The parallel mode causes multiple jobs to be submitted to perform CLRPFM operation: each job deletes the allocated range of records from the target file member.

Warning: Some of the ILE languages (COBOL, C) can invoke implicit CLRPFM commands at runtime, e.g. when a physical file is opened for OUTPUT. iSTREAM CCT provides no transformations 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).

CLRPFM (and similar) commands used for tables with CLOB/BLOB columns are also transformed, but executed using SQL DELETE command. Consequently,

- SQL_FAST_DELETE_ ROW_COUNT query option (QAQQINI table) may have to be used in order to prevent the SQL engine from implicitly using CLRPFM command
 - performance of the replacement clear process may not be as good as for the regular database tables, especially in the case of parallel clear
 - the above applies to all of the commands enabled for rollback, because internally all use the iSTREAM version of CLRPFM processor.
-

Records from the members being cleared are deleted either in a separate activation group (CRCV) or separate batch jobs. If job-level commitment control status is changed (CC is started or ended) iSTREAM mode for the job must be re-started (ENDISTMOD and then STRISTMOD).

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 job description or, if it cannot be located via *LIBL, with *LIBL/QBATCH job description.

SQL DELETE processing may or may not be recoverable by rollback depending on the setting of SQL_FAST_DELETE_ROW_COUNT parameter of QAQQINI file. Obviously, if an unrecoverable DELETE statement is executed, it is impossible to later rollback the target file using the rollback based on IBM journals. There are two options available to avoid this. The first is to set SQL_FAST_DELETE_ROW_COUNT parameter of QAQQINI file to *NONE. This is a very straightforward solution, but it may have undesirable performance implications at runtime. The second involves replacing 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 CCT-native CLRPFM. 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 iSTREAM RGSCLRPRC line command.

If all data modification operations for the unit are performed in iSTREAM mode, RMVJRNCHG command can later be used for rollback.

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

5.1.2 Object "protection" setup

In order to be able to rollback a CL command or a process using it by journal, commands must be enabled for rollback and STRISTMOD for unit must be used to define control library, list of "protected" libraries and unit journal. Obviously, in order to rollback data modifications the related objects must be journaled to the unit journal with *BOTH option.

CCT Rollback provides "rollback-to-checkpoint" functionality for files and data areas. When option 3 of iSTREAM is installed, the following commands found via the library list of the job performing the installation are enabled for rollback:

- CLRPFM
- CPYF
- CPYFRMQRYP

iSTREAM Command Transformer (CCT) Guide

- CPYFRMSTMF
- CPYFRMIMPF
- CPYSPLF
- DSPFD
- DSPOBJD
- RUNQRY
- DSPTAP
- DSPOPT
- RGZPFM
- CRCVRJC (in ISTSSYS)
- CRCRRJC (in ISTSSYS)

No other commands can be enabled for rollback, but if copies of these commands reside in other libraries, they can also be enabled for rollback. If CLRPFM command is disabled for rollback, most of the other commands will be transformed by iSTREAM but, since they are dependent on CLRPFM, will not be protected for rollback.

Warning: RGZPFM command in iSTREAM mode, if enabled for rollback transformation, is analysed by the command processor and an algorithm based on CPYF command is used for the actual reorganisation. The file data is copied to QTEMP library, then updates to all indexes (logical files) based on the file being reorganised are suspended, all records are deleted from the file, and the data is copied back from QTEMP. All indexes are then rebuilt.

This algorithm is used unless any of the following is true:

- KEYFILE parameter has a value other than *NONE
- RBDACCPH parameter has a value other than *YES
- ALWCANCEL parameter has a value other than *NO
- SRCOPT parameter has a value other than *SAME
- LOCK parameter has any value other than *EXCL
- FROMRCD parameter has any value other than *START
- RCDfmt parameter has any value other than *ONLY
- The target file is defined to not reuse deleted records (REUSEDLT(*NO))

If any of the above conditions are true, iSTREAM either ignores RGZPFM command, so that the rollback by journal still remained possible, or executes the standard RGZPFM (making rollback for the file impossible). The action taken depends on the iSTREAM system value RGZCPYFACT.

If the file being reorganised has multiple dependent indexes, they are removed and then sequentially restored. The process of rebuilding indexes (part of the restore) can be run in multistreamed mode. For that, SBMRCD command in ISTSSYS library has to be enabled for flash operations.

The iSTREAM version of RGZPFM command does not remove records marked as deleted.

For performance reasons it is highly recommended to use journal caching with the iSTREAM version of RGZPFM. If the performance impact of the iSTREAM RGZPFM mechanism is unacceptable, user-defined transformation (see section 4) can be used to either ignore the command or fall back on the standard IBM i functionality.

A file is protected for rollback in the following circumstances:

- An unqualified command from the rollback-enabled command list is entered either from the command line or from a program (commands entered using the command prompter are ignored by iSTREAM and no protection is provided in this case)
- The file resides in one of the protected libraries and is journaled to the journal defined for the unit with option *BOTH
- The last 7 characters of the description of the target file (table) member do not contain "ISTEXCL" value. The latter can be used for individual exclusion of files in protected libraries from protection.

5.1.3 *Journal checkpoints*

CRTCKPRLB (or CRTCKP) command creates journal checkpoints in the journal specified on STRISTMOD command. A checkpoint is a symbolic name given to an entry in the journal; it can later be used for journal rollback to the given entry. Checkpoints can only be created by jobs executing in iSTREAM mode and having both the list of "protected" libraries and rollback journal defined by STRISTMOD command. When a checkpoint is created COMMIT operation is used by iSTREAM, so if the job is running under the job-level commitment control definition all pending DB updates are committed too. This default behaviour can be changed using FCOMMIT parameter of CRTCKPRLB command.

5.1.4 *Cleanup*

iSTREAM CCT 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 rollback to any of the defined checkpoints from the library unit journal receiver library.

5.1.5 *Reporting*

The log of the rollback command execution is saved to RLBRPTunt file in ISTVQS library. A short version of the report is sent to a print file.

5.1.6 *Clone command option*

The rollback object protection functionality is based on the use of QIBM_QCA_CHG_COMMAND exit point of the IBM command processor. All of the protected commands are registered with this exit and whenever any of them is executed, the iSTREAM rollback protection processor is invoked.

This approach, however, has a few notable downsides. IBM documentation specifically says that a command cannot be transformed (i.e. protected) if

- The command was qualified with a specific library name, or
- The command has a parameter defined as RTNVAL(*YES), which allows the command processing program (CPP) to return a value to the calling program, or
- The command has a parameter defined as either DSPINPUT(*NO) or DSPINPUT(*PROMPT), which does not allow the value to be shown in the prompt display or joblog, or
- The command was used in a program running in system state.

In all of the above situations iSTREAM protection will not work. Also, if a command is invoked for the prompter, the exit point is not called at all.

iSTREAM offers an optional utility designed to address some of the above limitations. ISTVCM4 command line utility can be used to create clones based on the related iSTREAM processors for the following system commands:

- CLRPFM
- CPYF
- CPYFRMQRYF
- CPYFRMSTMF
- CPYFRMIMPF
- CPYSPLF
- DSPFD
- DSPOBJD
- RUNQRY
- DSPTAP
- DSPOPT
- RGZPFM

Clone commands with the above names can thus be created in ISTSSYS2 library and, if ISTSSYS2 library were added to the top of the library list, iSTREAM command processors would be called without the IBM Registration Facility exit point being used at all.

iSTREAM clones of the above commands, however, are slightly different from the original commands in that they have additional hidden parameters required for iSTREAM internal processing. This, therefore, may or may not be a valid option for any given implementation.

ISTVCM5 command line utility can be used to remove the clone commands created by ISTVCM4 from ISTSSYS2 library.

Both ISTVCM4 and ISTVCM5 utilities require *ALLOBJ special authority to execute.

5.2 Rollback interfaces

The rollback function has two main interfaces. The first is low-level, where all rollback parameters, such as names of libraries to be rolled back, number of rollback streams, etc. must be explicitly defined using RLBTCKP command parameters. The other is high-level; using this high-level interface, it is possible to define standing rollback data for a library unit and then simply refer to it at the time of rollback. Therefore, important decisions, such as how many rollback streams to use, what files to omit, and what files to rollback in separate streams, can be taken at the system configuration, rather than rollback time.

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

iSTREAM Rollback for Library Units supports multistreamed rollback when multiple jobs are submitted to share the rollback workload.

5.3 Rollback configuration and function

5.3.1 Functional description

All objects (database tables and data areas) to be rolled back by iSTREAM must be journaled with option *BOTH to the same IBM journal. Obviously, journaling for the objects has to be started before updates begin.

iSTREAM Rollback 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 rollback. A checkpoint can be used to rollback a single library unit.

Consistent rollback can only be achieved if all IBM i processes updating objects from a certain library unit execute in iSTREAM mode.

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

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

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

Two types of rollback are defined: *NEW and *CONT. Rollback of the type *CONT is used when the already rolled back 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 iSTREAM rollback scenario is possible:

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

If checkpoints are unavailable, two other options could be used: rollback to a journal entry with the given number (the last entry to be rolled back) and rollback to date-time. In the latter case the desired time interval and the direction of search for the target journal entry, i.e. the last entry to be rolled back to, are defined using RLBTCKP command date-time parameters.

The rollback process generates a QPQXPRTF print report in the output queue of the job. More detailed information regarding the rollback process can be found in ISTVQS/RLBRPT*unt* file. It has the same format as the output file of the IBM RMVJRNCHG command. Each rollback stream generates its own member in the file.

If the rollback did not remove any changes, the report file may not be created by the IBM RMVJRNCHG command. In this case, a template iSTREAM file is used. The print report shows a single line with strings of asterisks in each column.

In order to use rollback functions, the user has to be authorised to RMVJRNCHG command object.

Warning: In both single-stream and multistream modes the actual rollback is performed using the IBM RMVJRNCHG command (parallel multithreaded rollback described in the next section is based on the native ILE processor and represents an exception). RMVJRNCHG command fails with the exception message CPF7069 when there are no journal entries to remove. This may not be the desired behaviour, especially when the rollback is executed in the multistream mode. If one of the stream jobs fails, a related inquiry message is sent to QSYSOPR message queue. If the only reason for the failure is the objects specified on RMVJRNCHG command having no journal entries in the specified range, the failure should be ignored and the processing allowed to continue.

If a process attempting rollback of the data areas fails with CPF7069 message due to the libraries containing no data areas, at least one data area should be created in one of the libraries being rolled back in order to avoid the failure.

5.3.2 *Parallel multithreaded rollback*

By default, rollback is performed using the standard IBM RMVJRNCBG command. This command, however, can be rather inefficient in cases when the database objects have been actively updated by multiple concurrent processes. For example, if an account table has been updated by multiple parallel batch processing streams, the standard rollback may take the same amount of time as the total runtime of all the processes participating in the update.

In such cases, parallel multithreaded rollback can significantly improve the efficiency of the rollback process. Parallel rollback can be defined using FLIST parameter of RLBTCKP command (see section 5.4). Each database object in the FLIST object set is rolled back using a separate operating system process (job). On top of this, a number of threads (from 1 to 99) can be defined for any such object. If the number specified is greater than 1, then the actual rollback is performed using a native ILE driver program rather than RMVJRNCBG command.

This type of rollback, while offering performance benefits, may be compromised by unique indexes defined for the file (table), referential integrity relationships and/or triggers. An example of a problem that could result from this is "duplicate index entries" error messages at rollback. In such cases it is recommended to use pre-rollback (ISTRUX01) and post-rollback (ISTRUX02) user exit programs to remove and then reinstate indexes and/or triggers. To define any of the two user exits a program with the relevant name should be compiled into the control library for the unit. Each of the programs has to have two parameters, a 10-character short name of the file (table) and a 1-character response code. The latter has to be set up on exit from the program to '0' for normal end or any other character to indicate a failure.

Warning: multithreaded rollback is not supported for files with the maximum record length exceeding 32,133 bytes and for tables including CLOB and BLOB columns. When recovery to the transaction boundary (CMTDBY(*YES)) is requested, this type of rollback is not supported either. In the latter case a warning message is usually generated in the joblog and the single-threaded mode of rollback is assumed. If an attempt is made to rollback a program-described file with the maximum record length exceeding 32,133 bytes, the trailing bytes may be reset to X'40'.

Multithreaded rollback creates the following objects in ISTVQS library:

- *ADFLunt* - contains the value of FLIST parameter
- *PRTFunt* - contains a report of the multithreaded rollback (by file and thread)
- *PRTEunt* - contains error report for multithreaded rollback (by file and thread)

ISTREAM Command Transformer (CCT) Guide

Additionally, the parallel rollback work library is created for each unit where multithreaded file rollback is used. The name of the library is *ISTRunt*, where *unt* is, as above, the name of the unit. The library contains multithreaded rollback work files and data areas.

For each file (table) rolled back using the multithreaded mode, a spool rollback report is created. The name of the report is the same as the name of the file being rolled back.

In order to use exit program functionality, ISTRUX01 and/or ISTRUX02 program objects have to be compiled into the of the unit.

Example of ISTRUX01 exit program code

```

PGM          PARM(&QFILE &RC)
/* ----- */
      DCL          VAR(&QFILE) TYPE(*CHAR) LEN(30)
      DCL          VAR(&RC) TYPE(*CHAR) LEN(1)
      DCL          VAR(&FIL) TYPE(*CHAR) LEN(10)
      DCL          VAR(&LIB) TYPE(*CHAR) LEN(10)
      DCL          VAR(&MBR) TYPE(*CHAR) LEN(10)

      CHGVAR      VAR(&FIL) VALUE(%SST(&QFILE 1 10))
      CHGVAR      VAR(&LIB) VALUE(%SST(&QFILE 11 10))
      CHGVAR      VAR(&MBR) VALUE(%SST(&QFILE 21 10))

/** OMPF section begin ----- */
      IF          COND(&FIL *EQ 'OMPF') +
      THEN(DO)
/* OMXXLF */
      DLTf        FILE(QTEMP/OMXXLF)
      MONMSG      MSGID(CPF2105)
      CRTSAVF     FILE(QTEMP/OMXXLF) TEXT('OMPF logical file +
      backup')
      MONMSG      MSGID(CPF5813 CPF7302)
      SAVOBJ OBJ(OM05LF OM07LF OM72LF) +
      LIB(&LIB)      +
      DEV(*SAVF)    +
      SAVF(QTEMP/OMXXLF)
      DLTf        FILE(&LIB/OM05LF)
      MONMSG      MSGID(CPF2105)
      DLTf        FILE(ISTVQS/OM07LF)
      MONMSG      MSGID(CPF2105)
      DLTf        FILE(ISTVQS/OM72LF)
      MONMSG      MSGID(CPF2105)

      ENDDO
/** OMPF section End ----- */

/* ----- */
      ENDP:
      SNDPGMMSG   MSG('User exit completed successfully.')
      CHGVAR      VAR(&RC) VALUE('0')
      /* Journaling will be ended */
      GOTO ENDPG
/* ----- */
      ERROR:
      CHGVAR      VAR(&RC) VALUE('E')
      SNDPGMMSG   MSG('User exit completed with ERRORS!')
/* ----- */

```

iSTREAM Command Transformer (CCT) Guide

ENDPG: ENDPGM

Example of ISTRUX02 exit program code

```

PGM          PARM(&QFILE &RC)
/* ----- */
DCL  &UNT    TYPE(*CHAR) LEN(3)
/* ----- */
DCL      VAR(&QFILE) TYPE(*CHAR) LEN(30)
DCL      VAR(&RC)   TYPE(*CHAR)  LEN(1)
DCL      VAR(&FIL)  TYPE(*CHAR)  LEN(10)
DCL      VAR(&LIB)  TYPE(*CHAR)  LEN(10)
DCL      VAR(&MBR)  TYPE(*CHAR)  LEN(10)

CHGVAR      VAR(&FIL) VALUE(%SST(&QFILE 1 10))
CHGVAR      VAR(&LIB) VALUE(%SST(&QFILE 11 10))
CHGVAR      VAR(&MBR) VALUE(%SST(&QFILE 21 10))

/** OMPF section begin ----- */
      IF      COND(&FIL *EQ 'OMPF') +
      THEN(DO)
        CHGVAR      VAR(&RC) VALUE(' ')
        RSTOBJ      OBJ(OM05LF OM07LF OM72LF) SAVLIB(&LIB) +
        DEV(*SAVF) SAVF(QTEMP/OMXXLF)
      ENDDO
/** OMPF section End ----- */
/* ----- */
ENDP:
      SNDPGMMSG  MSG('User exit completed successfully.')
      CHGVAR      VAR(&RC) VALUE('0')
      GOTO ENDPG
/* ----- */
ERROR:
      CHGVAR      VAR(&RC) VALUE('E')
      SNDPGMMSG  MSG('User exit completed with ERRORS!')
/* ----- */
ENDPG:      ENDPGM

```

It would also be possible to both backup and restore access paths in parallel. For that, STRISTMOD command should be used in ISTRUX01 enabling flash-copy functionality.

For the parallel rollback file journaling is usually ended by iSTREAM. It helps improve performance of the rollback process. If ending file journaling is undesirable, ISTRUX01 exit program should be implemented and return response code '1'.

If rollback to a transaction boundary is requested, parallel multithreading parameters are ignored and the related file is rolled back using a single thread.

5.3.3 Rollback in High Availability Environments

The only high availability software currently supported is Precisely MIMIX HA. Furthermore, iSTREAM is only fully compatible with MIMIX for Power HA and MIMIX Global editions of MIMIX.

In order for the Rollback for Library Units Licensed Program to successfully operate in the MIMIX environment, the object and the file entry for *ISTVCKPunt* multi-member file must be added to the primary group definition. Further on, after *STRISTMOD* command is executed for the first time for a library unit, it is strongly recommended to make sure that the checkpoint 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 could be multiple approaches to rollback in high availability environments.

Recovery from both logical (program or data) errors and server failures

- Checkpoints are taken using *CRTCKPRLB* command with *HAMODE(*YES)* parameter
- After the failure MIMIX replication is suspended (DG is stopped)
- The required unit (source or target) is rolled back to the required checkpoint
- If both systems are still available, rollback could be executed on both systems
- MIMIX replication is cold-started

To make the above synchronous rollbacks in HA environment possible, *CRTCKPRLB* command, when executed with *HAMODE(*YES)* parameter, waits for MIMIX backlog to disappear before taking the checkpoint.

Recovery from logical (program or data) errors only

Approach 1

Checkpoints are taken in any mode

- After the failure MIMIX replication is suspended (DG is stopped)
- The source unit (source or target) is rolled back to the required checkpoint
- MIMIX is used to resynchronise the source and target libraries
- MIMIX replication is restarted

Approach 2

Prior to starting the batch, e.g. End-of-Day (EOD) process *MIMIX HLDDGLOG* command is executed for the replication data group.

- If the EOD process fails, MIMIX replication is not suspended for the data group and rollback is executed only for the source unit. MIMIX then replicates the changes to the backup system. After the EOD process ends *RLSDGLOG* command is executed for the data group. As *RLBTCKP* process ends journaling of the unit checkpoint file, it may cause MIMIX warning messages to be issued.

Warning: No rollback 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 can only be rolled back 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).

Ability to perform a rollback for a library unit may be compromised by such file level operations as Sort (FMTDTA command), Initialise Physical File Member (INZPFM command), or DSPxxx OUTPUT(*OUTFILE) commands. If any of these commands have been executed for any file in protected libraries the file automatically becomes unprotected and has to be omitted when RMVJRNCHG rollback is attempted.

Generally, files with LOB fields cannot be rolled back using CCT. 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 rollback is executed.

In order to use Approach 2 in scenarios with the parallel multithreaded file rollback, it is necessary to code ISTRUX01 exit program to return response code '1' instructing iSTREAM not to end journaling for the related files.

5.4 Menus and commands

Option 3 facilities can be accessed via "Rollback" menu group of the of the iSTREAM Access for MS Windows GUI or URLBK system menu. The latter has a slightly richer set of features mainly related to native command prompting.

"Define Advanced Rollback" menu element or option 1 "Enable advanced rollback for unit" of the 5250 menu creates iSTREAM standing data for the library unit. In order to do this, iSTREAM runs the user through a mock unit rollback scenario executing STRISTMOD, CHKUNTJRN, CRTCKPRLB, CHKOBJLCK, and RLBTCKP commands in prompt mode. The entered parameter values are stored in work structures in ISTVQS library. It is possible to make the system skip CHKOBJLCK step during the advanced rollback by setting OBJLIST parameter value to *NONE on CHKOBJLCK command during the mock rollback process.

Also, work management structures for the rollback process can be created. The list of such work management objects includes CRCVSBS*unt* subsystem, CRCVSBS*unt* job queue, and CRCVRJDxxx job descriptions; the subsystem description and the job queue are saved in ISTVQS 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 RLBTCKP command. At the execution time the first $\max(n+1,10)$ rollback streams are, therefore, run in separate main storage pools. Additional job CRCVRJDxxx 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 rollback configuration has to be redefined each time a new release of iSTREAM is installed.

Menu element "Advanced Rollback to checkpoint" or option 2 of the 5250 menu performs a simplified rollback using CHKOBJLCK and RLBTCKP commands. In order to execute RLBTCKP, its parameter values are extracted from the earlier standing data structures. In the 5250 mode there are two sub-options available: prompted rollback, when the end user can change some of the earlier defined parameter values and press Enter key in order to let the process continue, and automatic rollback, when the related commands are executed with no user intervention. The iSTREAM Access for MS Windows GUI only supports automatic rollback.

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

Menu element "Rollback to Checkpoint" or option 4 of the 5250 menu is the interface to the low-level unit rollback interface, RLBTCKP command. Most of its parameters are self-explanatory; help text is also provided. A few general comments, however, are relevant.

"Number of rollback streams" parameter is used to define the MINIMUM number of rollback streams requested. The actual number is calculated on the basis of the number of objects to be rolled back and explicitly specified additional rollback streams. Additional rollback 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 DSPORT DETAIL(*BASIC) system command. Such files are not rolled back themselves; they are used to contain the names of objects to be rolled back. The value of LIBLIST parameter in this case has to be set to *OBJECTS. This feature can be used for selective rollback, especially when the number of the files to be omitted from the rollback process is too high for them to be specified using OFLIST parameter.

iSTREAM rollback for library units option includes the following user commands:

- ENACMDRLB - enable command for rollback. This command can be used to selectively enable the commands from the list in section 5.1.2 for rollback. This command has OPTION parameter reserved for iSTREAM programs and service representatives.
- DISCMDRLB - disable command for rollback. This command can be used to selectively disable or disable the commands from the list in section 5.1.2

iSTREAM Command Transformer (CCT) Guide

for rollback, If a command is disabled for rollback and is then used with the target file that is part of a rollback group, later rollback attempts may fail. This command has OPTION parameter reserved for iSTREAM programs and service representatives.

- CRTCKPRLB (or CRTCKP) – define a journal checkpoint that could be used for future rollback.
- CLRCKPRLB (or CLRCKP) – remove a single checkpoint or all previously defined checkpoints.
- CLRRCV – remove all unused journal receivers (journal receivers containing no checkpoint entries or entries for superseded checkpoints only)
- LSTCKP - display, print or output to a file a list of defined checkpoints
- QRYCKPVRS - display, print or output to a file a list of versions of the specified control point available in *CURCHAIN of the unit journal
- PRSCKPRLB - create a symbolic link to the journal entry that could later be used to specify the rollback target for RLBOCKP and RLBOCKPA commands. The link can be created for a journal entry with a given number or a point in time.
- RLBOCKP – rollback to checkpoint. Prompting RLBOCKP always retrieves values saved during the latest mock rollback setup for the unit (see RLBOCKPA command description). The command generates a report saved to RLBRPTunt file in ISTVQS library, while a short version of the report sent to printer.

Rollback is supported to named checkpoints, journal entries or specific date-time points. Successfully completed rollbacks can be continued to earlier checkpoints (or their versions), journal entries or points in time.

If parallel rollback option is used, WLCGRP parameter of the command can be used to allocate the service processes submitted as separate jobs to pre-created a workload capping group. This could help reduce the impact of the rollback operation of the rest of the system in the partition.

- RLBOCKPA – invoke advanced rollback for library unit.
- ENARLBUNT – run the mock rollback process and create CCT rollback standing data. This command takes the user through a mock rollback scenario where the user defines journaling for the unit, creates a checkpoint, verifies object locks and rolls the unit back to the defined checkpoint. Parameters of RLBOCKP command used for the mock rollback are stored and then used when RLBOCKPA (Advanced Rollback to Checkpoint) command is invoked for the given unit. Execution of RLBOCKP command from the 5250 interface does not change the stored parameter values. Execution of RLBOCKP from the iSTREAM Access for MS Windows GUI, however, does change the stored values so that any subsequent invocation of RLBOCKPA, irrespective of the interface, makes use of these stored values. Optionally, RLBOCKP can be used to only save the

parameters entered and not perform the actual rollback during the rollback mock process. SKPRLB(*YES) parameter helps achieve this during the no-intrusive mock rollback.

Although technically possible, using advanced rollback for rollbacks to time-date and specific journal entries is not recommended.

- **CHKUNTJRN** – check and/or start journaling for the library unit objects. It is recommended to use this command before entering iSTREAM mode for the unit for the first time. CHKUNTJRN 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. If CHKUNTJRN is executed as part of a mock rollback (see the description of ENARLBUNT command), parameter values used are stored and then retrieved for the advanced rollback (RLBTOCKPA). If the command is executed from the iSTREAM Access for MS Windows GUI, even outside of the mock rollback procedure, the stored values are also changed to those used on the command. Prompting CHKUNTJRN always retrieved values saved during the latest mock rollback setup for the unit.

The special value *PRMLST of LIBLIST parameter refers to the list of "protected" libraries specified on the latest STRISTMOD command executed for the unit. If the command is used with the unit name and the value of SAVREF parameter is set to *YES, parameter values used are stored and can be later retrieved using the special value *UNIT.

- **CHKOBJLCK** – verify existing locks for physical files and data areas. It is strongly recommended to run this verification before each rollback operation (RLBTOCKP). Locks held by the job where CHKOBJLCK 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.

Special value *PROTECTED of LIBLIST parameter refers to the list of "protected" libraries specified on the latest STRISTMOD command executed for the unit. If the command is used with the unit name and the value of SAVREF parameter is set to *YES, parameter values used are stored and can be later retrieved using the special value *UNIT.

CHKOBJLCK command can be invoked using its proxy command objects VFYOBJLCK.

- VFYFDLCK command introduced in earlier versions is also available but is not used by the automated rollback facility; therefore, automated rollback configurations created by iSTREAM releases earlier than V3R5M0 have to be recreated by executing ENARLBUNT command - otherwise, the rollback may fail or run into locking problems. The special value *PRMLST of LIBLIST parameter refers to the list of "protected" libraries specified on the latest STRISTMOD command executed for the unit. If the command is used as part of the ENARLBUNT process, the list of libraries is stored and can then be referred to with the special LIBLIST value *UNIT.

- RGSCLRPRC – to register CRCVSCL stored procedure (CLRPFM)
- DRPCLRPRC – to drop the earlier registered CRCVSCL stored procedure
- CRCVRJC and CRCRRJC commands are used during the rollback process and must always be enabled for rollback. In order to prevent inadvertent removal of these command objects from the list of commands enabled for rollback ENACMDRLB, DISCMDRLB and LSTCMDRLB functions have two modes of operation, *USER and *SERVICE. The latter mode is used by iSTREAM programs, the former is the default mode. Users can override this mode by specifying OPTION(*SERVICE) parameter for the above iSTREAM commands.

5.5 Exits defined

Rollback facility optionally makes use of QIBM_QCA_CHG_COMMAND exit points with number starting from 1882110101 (maximum 100 commands can be enabled for all types of transformation). Flash Operations (option 1), Command Transformation (option 2) and Command Rollback (option 3) share the command object space and exit points. 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. The 10-character data area must contain the first number of the range to be used by 7S77STR installation exit program minus 100 (by default, the number used is 1882110001).

In addition, Rollback facility makes use of QIBM_QCA_RTV_COMMAND exit points with numbers in the 1882110201-1882110230 range.

6. CL command transformation algorithm

When a CL command is enabled for either flash (asynchronous) execution, configurable transformation, or rollback, QIBM_QCA_CHG_COMMAND exit is registered for this command, so that each time the command is about to be executed, the exit program is invoked. The exit program of iSTREAM then makes use of the following procedure in order to process the request.

1. Verifies whether transformation is allowed by the IBM command processor and whether the command is being invoked in the prompt mode. If transformation is not allowed or the command is being prompted, no further processing takes place. The command is returned to the IBM command processor unchanged.
2. Verifies an external (to iSTREAM) exit program definition for the CL command (CRCVSAR file in ISTVQS data library can be used to define those). Since only one exit point can be defined for any given command object, CRCVSAR facility could be used when at the time of iSTREAM implementation a new iSTREAM-defined exit point needs to be created for a command that already has one defined. In such cases, for compatibility purposes the existing exit point should be deregistered and the program earlier defined as the exit point – redefined as a CRCVSAR exit. If such exit is defined, control is given to it before any of the iSTREAM components attempt transformation and the output is passed to the next step of the procedure. The parameters passed to the command analyser user exit defined in CRCVSAR file have exactly the same format as stipulated by QIBM_QCA_CHG_COMMAND exit. Otherwise, the original command is processed by the next step. If a negative value is returned by the CRCVSAR exit program as the command length parameter value, all further processing of the command, including transformation, flash-execution and rollback substitution, is skipped. CRCVSAR can be used to define up to 50 external exit points.

Every time CRCVSAR file is updated, CRCVCSA utility program has to be called to update the CRCVSAR cache.

3. If the command is qualified, no further action is taken and the command is returned to the IBM command processor.
4. iSTREAM-configured command transformation is processed for the command. It includes verification of whether the command is enabled for configurable transformation and also whether this type of transformation is enabled in the current unit.
5. The output of step 4 is verified for rollback transformation enablement. It includes verification of whether the command is enabled for rollback transformation and also whether this type of transformation is enabled in the current unit. If enabled, the target file member is cleared using CCT native "CLRPFM" facility, the command string is transformed and passed to the next step.
6. CL command is tested for flash(asynchronous) execution enablement. It includes verification of whether the command is enabled for asynchronous execution and also whether this type of transformation is enabled in the current unit. If asynchronous transformation for the command is enabled, it is submitted for flash(asynchronous) execution and DUMMY command is

iSTREAM Command Transformer (CCT) Guide

returned for execution in the current job. Otherwise, the command is returned for execution in the current job.

If there is a requirement to add an application Command Analyser exit for a command that requires QIBM_QCA_CHG_COMMAND exit point for the command to be defined in iSTREAM, it is possible to use CRCVSAR external Command Analyser exit registration facility in iSTREAM. In this case only iSTREAM exit should be defined for QIBM_QCA_CHG_COMMAND exit point for the command, with the application exit program being defined in CRCVSAR file in ISTQS library. This file is created when iSTREAM version 3 or higher is installed with each record of the file identifying the related CL command, the exit program and the library it resides in. DFU can be used to add and manipulate information in CRCVSAR file.

When iSTREAM is deleted all CRCVSAR exit points must be re-registered using ADDEXITPGM API.

7. iSTREAM CCT restrictions

Modifications to system commands enabled for flash execution or rollback, such as changing their names, replacing them with commands having different parameters, etc. may cause iSTREAM CCT to fail.

If multiple versions of the same system command exist in different libraries and their default parameter values are set up differently using CHGCMDDFT system command, it may also cause iSTREAM CCT to fail. In such cases the recommendation is to create proxy commands rather than having multiple versions of the same command in different libraries. Proxy commands are fully supported by iSTREAM CCT and can be created using CRTPRXCMD system command.

Appendix A. STRISTMOD command notes

STRISTMOD is the command that can be used by an IBM i job to enter the iSTREAM mode of execution. In some cases, e.g. for unit '@@@', this command is issued implicitly by core iSTREAM programs.

STRISTMOD command serves three purposes:

- it starts the iSTREAM mode for the job where it is executed thus enabling relevant iSTREAM functionality
- it defines a subset of iSTREAM functions to be enabled in the current job
- it saves the entered parameter values in common storage thus setting up a unit, i.e. a group of jobs sharing the same iSTREAM configuration.

A job can either enter the iSTREAM mode with the earlier defined set of parameters (special values *UNIT) or redefine some (all) of them. While it is possible to change the unit parameters while other jobs of the unit are executing, this is generally not recommended because of unpredictable side-effects.

STRISTMOD parameters belong to several functional groups.

Common parameter group includes one parameter:

UNIT - name of the unit being created, joined and/or redefined.

Flash execution parameter group also includes parameter:

ASYEXE - definition of the flash execution functionality for the jobs executing in iSTREAM mode.

ASYDDM - DDM file used to submit flash (asynchronous) processes to a remote partition. If this parameter is defined, all flash requests are submitted to the related remote partition.

WLSASY - name of the workload capping group that background flash processes are assigned to.

Likewise, User-defined command transformation group includes a single parameter:

CMDTFM - a trigger enabling or disabling the entire user-defined command transformation functionality for the job(s).

Rollback group includes the following parameters:

CTLLIB - name of the control library, i.e. the library where the checkpoint file is created. This parameter can only be used if the unit journal has also been defined.

JRN - name and library of the unit journal. This journal is used when a new rollback checkpoint is defined. It is also used for subsequent rollback. A journal can be used without the control library. If the latter is not defined, no checkpoints



can be taken or used for the actual rollback, but all other rollback functionality can still be used.

LIBLIST - the list of libraries used at runtime for rollback object protection (see sections 5.1.1 and 5.1.2). If JRN parameter is defined, only objects journaled to this journal are protected. If JRN parameter value is *NONE but LIBLIST is defined, all journaled objects from the libraries of the list are protected.

Special value *PRMLST can be used to refer to the permanent library list for the unit that can include up to 200 libraries. In order to create this unit the libraries have to be added to a member of a source file and then compile using CMPPRMLST command. The name of the member has to be the same as the name of the unit the list is built for.

This list does not change when STRISTMOD command is executed, but the latter can be used to temporarily disconnect and then again reconnect the list to the unit.

If the journal is not defined for the unit, then any journaled file in a library included in LIBLIST is protected. If the journal is defined, files in libraries from LIBLIST are protected only if they are journaled to the unit journal.

NREC - the number of records to be allocated to each "record delete" stream in rollback protection scenarios. Setting NREC parameter to the special value *NONE instructs iSTREAM to always synchronously delete all records of each file in the main job.

AUTOJRN - controls the method of starting journaling for the checkpoint file created when STRISTMOD command with the given value of CTLLIB parameter is executed for the first time. iSTREAM can either start journaling of the newly created file or let other processes, e.g. MIMIX, to do it instead.

User-defined Multistreaming group includes the following parameters:

HOTLIB - name of the "hot" library for the unit. For more detail please see the iSTREAM Generic Multistreaming Toolkit manual.