RC INFORMATION	class EXT	repl.		ident CAP 790802	
	X RC 4000	X RC 6000	X RC 8000	RC 3600	page 1/2

subj. TRANSFORMATION OF ISQ AND CF MASTER FILES FROM RELEASE 10 TO 12.

A new utility program, newhead_isq, is included in the System Utility Package, SW8010/1 release 11.0. The program belongs to the Backing Storage Subpackage, which has the releasenumber 12.2, and its purpose is to transform isq and cf master files from the format used in release 10 to the format used in release 12.

call :

<new_file> = newhead_isq <old_file>

newhead_isq <file_1> <file_2> <file_3> ...

function :

1. If <new_file> is specified as leftside in the call, an existing <new_file> with one of the scopes temp, login, user og project is used. If <new_file> does not exist or it has scope system or undefined scope, a temporary <new_file> is created on the disc haveing the most free segments. The transformed file is placed in <new_file> and <old_file> is left unchanged.

2. If no leftside is specified in the call, the rightside files are all transformed from version 10 to version 12. The files must have one of the scopes temp, login, user or project.

The catalog tail elements 7, 8 and 10 are kept unchanged from the old file. Short-clock is written into tail element 6 and contents-key into element 9 as defined by the CF/ISQ system.

requirements :

process size : min. core size is 15000 hw. 30000 hw or more is recommended.

work resources : if no leftside exists in the call, a temporary workfile with a size equal to the biggest of the rightside files is created.

result :

transformation ok : ok.yes warning.no

transformation not ok : ok.no warning.yes

error messages :

In case one or more of the following error messages occur, nothing in the files has been changed.

1. leftside exists :

***newheadisq no core

- no input file
- too many parameters
- name conflict
- input file name illegal
- input file does not exist
- input kind
- input not isq file
- filehead not version 10
- output file used by other process
- output file extension impossible, claims exceeded
- output file creation impossible, claims exceeded

2. no leftside exists :

***newheadisq no core

- file name missing
- <file name> name illegal
- <file name> does not exist
- <file name> kind illegal
- <file name> scope system
- <file name> scope undefined
- <file name> no work resources
- <file name> not isq file
- <file name> filehead not version 10
- <file name> duplicate name

normal output :

During processing file name, number of records and the type of the file is printed.

RC INFORMATION	class EXT.	repl.	ident EAH810401
	X RC 4000	X RC 6000 X RC 8000	RC 3600 page 1/2

subj. Handling of ISQ-files in a Coroutine System.

1. The problem.

A dangerous pitfall exists when the ISQ-system is used in a coroutine system with implicit passivate on the isq-zone.

A runtime error in an activity is often trapped (by activate or by a trap label in the monitor block or an outer block) with the intention of closing the files before leaving the program.
You will then probably call the procedure setreadi to remove a possible update mark from the ISQ-file.

If, however, the alarm occurred while another activity was implicit passivated during an ISQ-operation, this operation is never finished.

Suppose the unfinished operation was a call of getreci. This procedure begins by saving the keyfields of the wanted record, after which the corresponding block is read into the zone, and at this point the activity is passivated.

Whenever setreadi is called on a file in update mode, the contents of the saved keyfields are restored into current record in order to prevent the user from destroying the keyfields. But current record is still the previous one because the last call of getreci hasn't been finished. So the keyfields of the wanted record are inserted into the old current record causing an erroneous key sequence in the file.

2. Remedy.

After a run-time alarm in a coroutine system you should always allow the implicit passivated activities to finish their started area transports.

The following procedure will do the job when called in the monitor block, provided that the coroutines are passivated by something else than area transports.

```
external
procedure finis_trans;
begin
  comment    the procedure finishes area transports in implicit
             passivated activities;
  integer array ia(1:12), messbuf(1:3), proc_descr(1:1);
  integer max_act, act, res;

  max_act:= system (12, 0, ia);

  for act:= 1 step 1 until max_act do
  begin
    repeat
      system (12<*act.descr*>, act, ia);
      res:= ia(8);
      if res = 2 then  <*implicit passivated*>
      begin
        system (5<*core move*>, ia(1), messbuf);
        if abs messbuf(3) > 100 then  <*not pending answer*>
          system (5, abs messbuf(3), proc_descr)
        else
          proc_descr(1):= 4;
        res:= if proc_descr(1) = 4  <*kind=area*>
              then activate(act) extract 24
              else 0;
      end res = 2;
    until res <> 2;
  end for act;
end finish_area_transports;

end;
```

The procedure Removeupdi has been changed to allow the parameter of type string as well as type zone.

When called with a zone as parameter, the effect of the call is exactly as described in RCSL No 31-D635: Corrections to RCSL No 31-D601. When called with a string as parameter:

Call: remove_upd_i (s)

remove_upd_i (return value, integer). Result of call:
0: The file contains no update mark.
1: The file contained an update mark which is now removed.

s (call value, string).
Specifies the file.

Function: Removes a possible update mark from the file.

To accomodate the extension the following runtime alarms have been added to the procedure:

param	Illegal type parameter to removeupdi.
lookup <i>	The file could not be looked up. <i> = result.
kind <i>	The file is not an area. <i> = kind.
contents <i>	The file is not an isqfile (<i> = 22).
reserve <i>	The file could not be reserved. <i> = result.
status <i>	Error during input/output to the file. <i> = decimal status.

information	repl.		ident FB 821018		page 1/1
		RC8000			class EXT

subj. ISQ Procedures and Zone State

Backing Storage release 13.2 introduces a change in order to make the system work properly when allowing implicit passivate during transports in ISQ-zones.

Some ISQ procedures assign a default value to "resulti" at entry and reassign at exit, possibly after passivation, only if the result deviates from the expected. In the meantime another coroutine may have changed the value of "resulti".

The problem has been overcome by letting the ISQ procedures save "resulti" in the leftmost halfword of the zone descriptor variable "zone-state" during the period of possible implicit passivation (I/O-operations).

The change will trap concurrent I/O operations in the same ISQ-zone executed by more coroutines.

A zone state alarm, however, will occur if user programmed routines (trap-routines and block-procedures) are activated and allowed to call high level I/O procedures before the original ISQ procedure restores the situation to normal. The problem may be prevented by assigning (getzone and setzone) the rightmost halfword of "zone-state" to "zone-state", i.e. "zone-state:= zone-state extract 12;".

RCSL No: 42-i 1321