Smart/RESTART

z/OS Application restart made simple

Overview

Abends and downtime can no longer be tolerated as organizations progress towards `twenty four by seven' operation. Smart/RESTART conserves your precious batch window by letting you ***resume*** an abended job from a checkpoint rather than ***rerun*** the job from the beginning. Without complicated restart logic or cumbersome operational procedures. Often without changes to source code or JCL.

In a shared data environment, checkpoint / restart capability is not just insurance but a *prerequisite* for concurrency. Smart checkpointing releases locks and eliminates the need to restore Db2, MQ, IMS and VSAM data to their initial state – so updates to shared data made by concurrent processes are *not* lost.

Smart/RESTART makes it as easy as possible to make both new and existing applications restartable. Simply define the logical unit-of-work and Smart/RESTART does the rest. Many applications which implement a unit-of-work loop can run restartably *without* source changes. You can even restart-enable legacy applications that *take no checkpoints* by defining the unit-of-work on the basis of criteria like Db2 column values, record content, I/O counts and/or SQL accesses.

How it Works

Smart/RESTART extends the scope of a unit-of-work to guarantee that changes to Db2, MQ, IMS and other RRS compliant resources stay in sync with your program's sequential file and cursor position, working storage and random VSAM updates. This enables your applications to restart after abends, recompiles, even system IPLs – with *all* resources in a consistent state.

Smart/RESTART overhead is very low because Smart checkpointing is extremely efficient and its Repositionable Sequential Access Method's parallel I/O significantly outperforms QSAM.

Smart/RESTART is a robust, mature and proven product that is successfully implemented in Fortune 1000 organizations worldwide. RAI's knowledgeable and responsive technicians are available 24 hours a day, 7 days a week to keep your z/OS batch production running smoothly.

Benefits

- Update Db2, MQ, IMS, VSAM and other RRS compliant resources in the *same* application
- Restart from the last checkpoint (near the point of failure) or any prior checkpoint
- Restart ***automatically*** after deadlocks and timeouts
- Vary checkpoint frequency ***dynamically***
- Reduce application size and complexity
- Improve concurrency and availability
- Eliminate the need to develop restart logic separately for each application
- Allow failing applications to be corrected, compiled and relinked – and then restarted from a checkpoint
- Use the ***same*** JCL for both initial and restart runs
- Eliminate the need to back out and redo successfully processed work

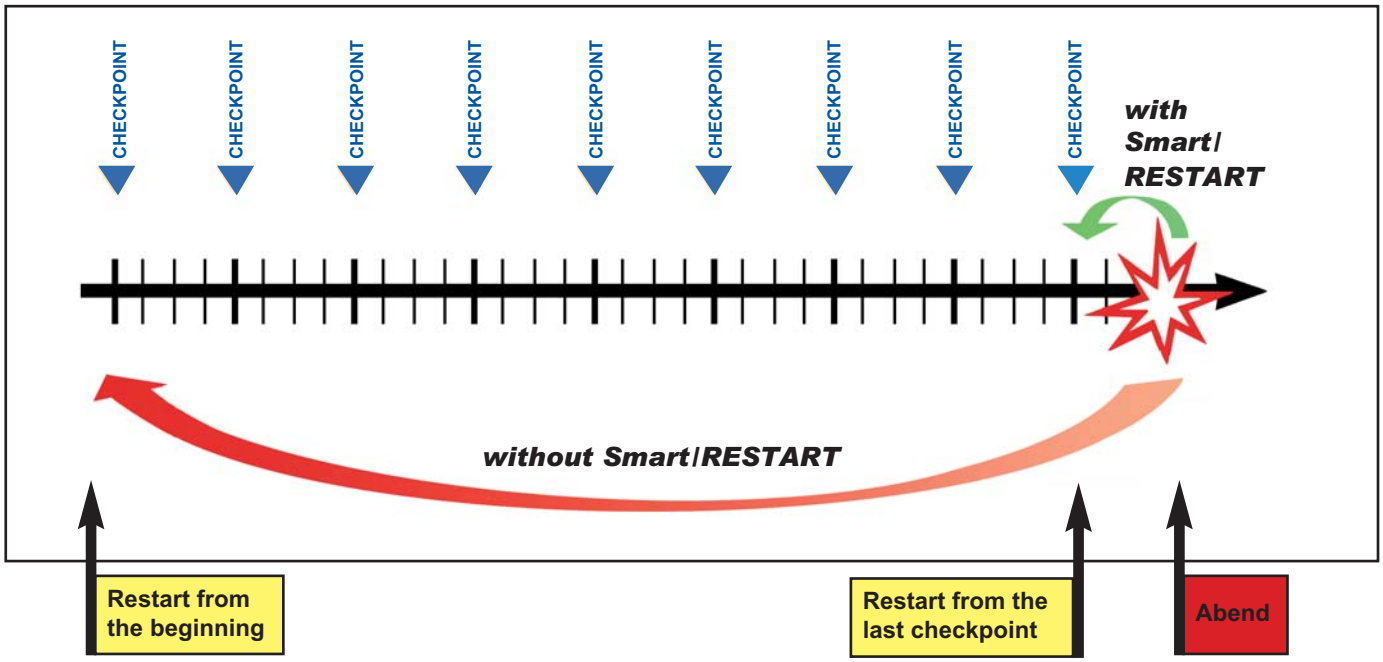
Features

- Reposition Db2 cursors automatically, without program logic
- Predefine different checkpoint frequencies for separate shifts
- Restart failing applications on any system in a Db2 data sharing group
- Edit program storage and repair bad data *during* execution
- Suspend the processing of one or more applications at points of consistency and resume them on command
- Gracefully cancel active jobs at unit-of-work boundaries
- Develop hybrid applications using COBOL, PL/I, C and/or Assembler languages
- Mix compile and runtime libraries *without restriction* for all COBOL, PL/I and LE release levels
- Use native COBOL verbs like OPEN, READ, WRITE, CLOSE and START to access repositionable files and VSAM datasets
- Access SMS compressed and extended format datasets without restriction

Riverview Historic Plaza • 33 Newark Street • Hoboken NJ 07030 USA • Tel: +1 201 420-0400 • www.relarc.com

Relational Architects International

With and without Smart/RESTART



Smart/RESTART JCL

The figure below shows a sample Db2 batch application run with Smart/RESTART and Smart/RRSAF (a separate product from Relational Architects). Use of Smart/RRSAF is optional but enables such services as automatic restart after Db2 resource unavailable conditions and automatic cursor repositioning.

Smart/RRSAF provides an alternative to running Db2 batch applications under the TSO Terminal Monitor Program and DSN / RUN commands. Smart/RRSAF lets you run your Db2 batch applications with regular JCL so they *look* and *behave* like standard OS jobs. Smart/RRSAF also permits granular SMF job accounting, rather than lumping charge-back statistics for all Db2 batch applications under the TSO TMP.

```
//EXAMPLE  JOB
//STEP1    EXEC PGM=EXAMPLE
//INFILE   DD DSN=REPOSITIONABLE.INPUT.FILE,DISP=OLD
//OUTFILE  DD DSN=REPOSITIONABLE.OUTPUT.FILE,DISP=OLD
//VSAMFILE DD DSN=RANDOMLY.ACCESSED.VSAM.KSDS,DISP=OLD
//RAINPUT  DD *   <-- Optional Smart/RESTART parameter file
  CKPT_PACE(50)
//
```

Prerequisites

Smart/RESTART operates with all releases of z/OS, Db2 for z/OS, WebSphere MQ and IMS whose IBM support status is current, as well as RRS compliant resource managers from other vendors.

Smart/RESTART Dialogs

Smart/RESTART includes Smart/MONITOR and Smart/PREDICTOR to help you monitor and control your restartable jobs and estimate their completion times. Smart/MONITOR lets you modify checkpoint frequency as well as suspend, resume and quiesce your restartable jobs. You can display statistical summaries of file I/O, SQL access and CPU utilization – even edit program storage and repair bad data.

Smart/PREDICTOR (below) graphically displays the progress of your job. Smart/PREDICTOR's completion time estimates enable operators to take remedial action to meet production schedules.

```
------  Smart/PREDICTOR Run Time Estimator  -------
Command ===> _

Run Time for job EXAMPLE
   Time Processed     ===> 01:20:48
   Time Remaining     ===> 02:52:30
   Estimated END time ===> 17:51

Percentage of Processing Completed
   |||||||||||||||>.....|.....|.....|
   0         25         50         75         100
          16% completed this run -
          66% completed since initial run
```