



1. If the application does COMMITs, the database can be left in an inconsistent state. If a long update has periodic COMMITs without being restartable, then in order to rerun the job these COMMITted updates will have to be backed out somehow.
2. If the program does not do any COMMITs, in order to complete the entire update process on the table as a single transaction, the application locks the table for update, preventing access to it by other programs.

In both cases, downtime recovering from an abnormally ended job can be considerable and even involve restoring a backup version of the database.

Adding restartability to an application adds reliability. The application can roll back the database to the last COMMIT point, and will know the database is in a known good state. But the COMMIT by itself is not enough if the application's state information itself has been lost. A restartable program has the ability to restore the application program itself to the state it was in at the last COMMIT point. With both parts of the equation restored to their exact state as of the time of the last COMMIT, the program simply picks up where it left off and continues executing.

Restartability has two aspects: recovery from abnormal terminations, and intentional stops and restarts. Although checkpoint/restart has traditionally been thought of as an error recovery tool, the ability to *intentionally* abort jobs and restart them can be very powerful. If a reboot is needed for maintenance, jobs may be stopped, and then restarted after the reboot. Jobs may also be stopped and restarted to balance load on machines. The possibilities are limitless.

## Designing Applications For Restart

To design a restartable application, first identify the *logical unit of work* (LUW). This is the main processing loop, or the transaction that the program is processing. A restartable program must have some transaction (or iterative processing) at its core. The application should be logically complete even without adding calls to the checkpoint API.

To initialize the restart functions of the checkpoint library, the application code calls the INIT function near the beginning of its run. We encourage designers who are new to restartable programs to follow this process in the sample programs which come with the checkpoint library.

The application program opens files using the supplied I/O functions. These functions save information about the open files so the checkpoint library can know the file position when it is time for a checkpoint. Provided are both COBOL compatible I/O functions as well as functions which allow the use of the C stdio library.

Note that file repositioning is intended for sequentially accessed files. Some files are not eligible for repositioning, such as sort work files. Also, temporary files should be used outside of the main transaction loop, and should not be repositionable files managed by the checkpoint API.

At the end of the LUW, just before a database COMMIT is taken, the application code calls the CKPT function to take a checkpoint. This takes a snapshot of the positions of open files, user memory variables, and so forth. If something happens to cause abnormal program termination, the program can pick up from the point of its last CKPT call and not have to start over from the beginning.

If the program does terminate abnormally, the next time it is run the INIT call detects that it has been restarted. It then restores the user storage and repositions the files. It returns a code to the calling program to let it know that a restart is in progress.

At the end, the user program calls the TERM call to tell the checkpoint library to clean up its tables and remove this job.

This manual contains descriptions of the checkpoint API calls, and examples of their use in both C and COBOL.

*Cautionary Note:* Restartable programs require attention to detail to work properly. Even though the checkpoint library provides the restart functionality, it is up to the developer using the checkpoint library to see that the guidelines in this manual are followed and that the program is designed to take advantage of restart features.