Published: 04/03/67

Identification

Delete a subtree of the file system hierarchy.
deltree
E. Q. Bjorkman

Purpose

Deltree is the procedure used by the delete command (BX.8.07) to delete an entry which points to a non-empty directory. In order to delete such an entry delete calls deltree to delete the subtree beneath the entry. The method used in deltree can be easily adapted for other tasks which are repeated at all nodes of some tree structure in the file system.

Usage

call deltree (path, failsw);

path is the path name of a directory. Failsw is a 1-bit switch indicating on return that some entry of path could not be deleted.

Deltree starts deleting entries of the tree structure beneath path at the end nodes (i.e., directories that have no directories inferior to them). Deltree reaches these end nodes by constructing a path name of a directory immediately inferior to path and then calling itself recursively with that path name until the end nodes are reached. When deltree has deleted all the entries in an end-node directory which it can delete, it returns to its caller.

Implementation

call deltree (path, failsw);

dcl path char(*), failsw bit (1);

Deltree first obtains the current calandar clock time using the PL/I built-in abnormal function "clock_". This time is used later to determine whether entries were added to the directory path after deltree started its work. Deltree then calls the Directory Supervisor primitive

list_dir (BY.8.02) to get a list of all the entries in the directory. The branches are operated on one by one in the order in which they appear in the directory; links are left till later.

The operation performed on a branch proceeds as follows. If the branch is not a directory, deltree calls delete_entry (BY.2.01) to delete the branch. If the branch cannot be deleted, deltree sets failsw equal to "1" b. Deltree moves on to the next branch whether or not the branch is deleted.

If the branch is a directory, deltree calls itself recursively with the path name of that directory as the argument path. When the recursive generation of deltree returns, deltree calls delete_entry to delete the branch. If the branch cannot be deleted, deltree sets failsw equal to "1" b; deltree moves on to the next branch in either case.

Eventually all branches, directory and non-directory, will have been operated on. In the ideal case path now has no branches. However, two types of branches may still be in path: those which could not be deleted and those which were added during the time deltree was operating on the contents of path.

Deltree again calls list_dir to obtain a list of path and again operates on each branch (if there are any), this time in a different manner. The date and time on which each branch was last modified is compared with the calendar clock time obtained right after deltree was called. If the branch was modified after that clock time, it may have been added while deltree was working on path. (It may also be an "old" branch which was modified in some way, but since no harm is done by operating on that branch again deltree assumes it was added.) Deltree attempts to delete those branches which may have been added, making no distinction between directory and non-directory branches and setting failsw equal to "1" b if the branch cannot be deleted.

When all branches have been checked, deltree deletes all the links in path (note that links can always be deleted) and returns.

Errors

Possible errors and the action taken by deltree are:

1. The user does not have the read attribute on in path (i.e., path cannot be listed).

This error simply means that deltree cannot obtain a list of the contents of path and consequently cannot delete all entries. The error is not signalled. Deltree merely sets failsw equal to "1" b and returns.

2. The segment path is not a directory.

Deltree records the error as outlined in BY.11.00 and signals the condition deltree_001. If the signal returns control to deltree, deltree returns to its caller.

Note that failure to delete a branch is not considered a serious error. Rather it is one of two possible situations (able to delete or not able to delete). Deltree simply notes the failure by setting failsw equal to "1" b and goes on to operate on the next entry.