CS 6210 Project 4 Recoverable Virtual Memory

README

Compile the code

1. Execute the make file using this command ‘make clean’.
2. Then, enter the command ‘make’.
3. ./rvm

Implementation Design

Figure 1 shows a high level overview of the design. We have the concept of a “Recoverable Virtual Memory Manager” that manages segments of memory. For each segment of memory, it keeps a back up on file. This backup facilitates persistence. If a transaction to a memory segment is committed then the back up file is updated. However, if a transaction is aborted, then the memory segment upon which it was performing the transaction is reverted with the back up file for that memory segment.

All committed changes are kept track of in a log file, which is synchronized on disk for persistence.

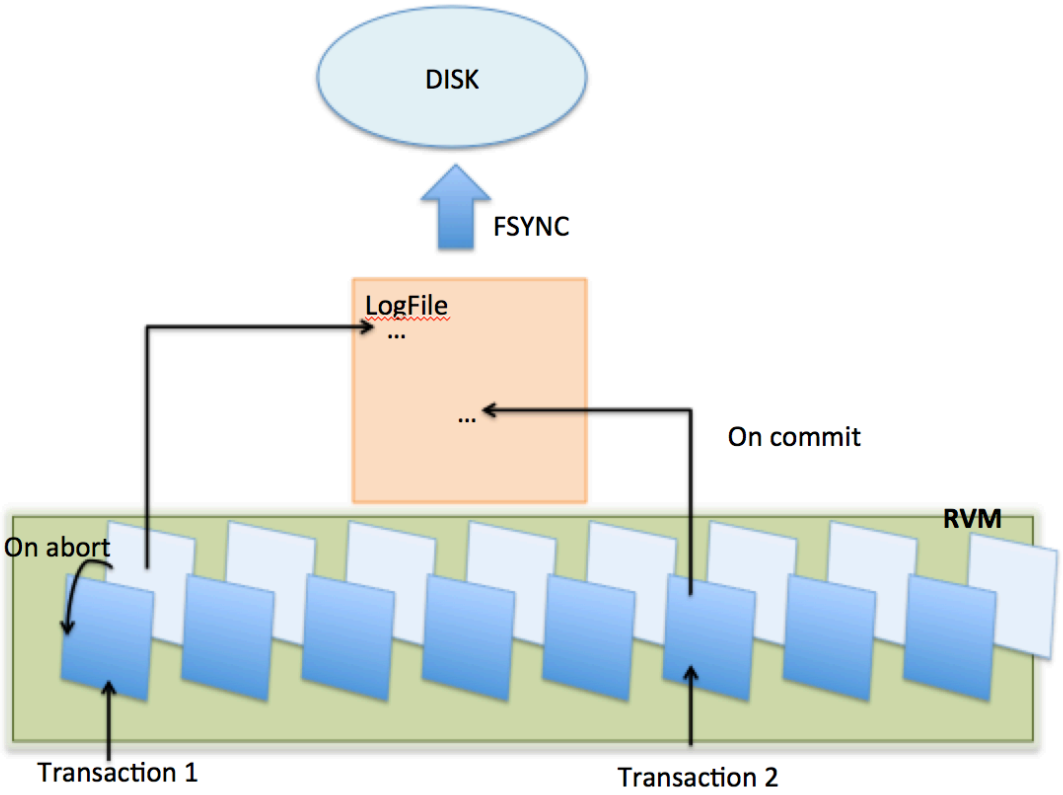

Figure 1: Recoverable Virtual Memory: On Commit and On Abort

Implementation Data Structures

Our implementation contains the following data structures:

Data Structure	Description
File_t	• The filename where the memory segment is persistently stored upon commit. • The size of the file. • The pointer to the memory segment. • The pointer to the previous version of memory upon soft updates. This is used at abort to revert changes.
Trans_range	• The offset in a memory segment for which the change occurred. • The size of the memory segment change.
Transaction	• The transaction id • The memory segment that the transaction has modified.
Rvm_struct	• The directory in which the file segments and log for an RVM instance is saved. • The file segments that are managed by this recoverable memory. • The transactions that occur on this RVM.

API Description

Function Name	Implementation Description
rvm_init()	Initializes a Recoverable Memory Manager object with the provided directory. Creates a log associated with this Recoverable Memory Manager.
rvm_map()	Maps memory segment of a given size to the recoverable memory manager.
rvm_unmap()	Unmaps memory segments from the recoverable memory manger.
rvm_destroy()	Destroys memory segments.
rvm_begin_trans()	Creates a new transaction object and registers it with the Recoverable Memory Manager whose memory segment it will be modifying.
rvm_about_to_modify()	Copies the memory segment about to be modified to a back up memory location.
rvm_commit_trans()	Saves the memory segments modified by the transaction to a file, which can be accessed by multiple processes.
rvm_abort_trans()	Reverts the modified memory segment to the content of the back up memory location.
rvm_truncate_log()	When a segment is destroyed the log entries pertaining to that segment are truncated. This is to ensure that the log file retains no entries pointing to a nonexistent segment.