GEORGIA INSTITUTE OF TECHNOLOGY

Recoverable Virtual Memory
CS 6210 Project 3 Report

Paras Jain, Manas George

1 PROJECT GOAL

The aim of this project is to implement a recoverable virtual memory system. This system allows users to manage peristent memory using a transaction-based API. This mechanism allows clients to create applications with persistant and consistent data structures (e.g. a database).

2 API

1. `rvm_init`: initializes the log files and directory for the VM system
2. `rvm_map`: maps a segment file on disk to area in memory
3. `rvm_unmap`: unmaps a segment from memory
4. `rvm_destroy`: cleans up the backing store for a segment
5. `rvm_begin_trans`: creates a transaction
6. `rvm_about_to_modify`: notifies library about intended modification to segment of memory
7. `rvm_commit_trans`: commits the transaction to disk
8. `rvm_abort_trans`: undo all modifications during a trasaction
9. `rvm_truncate_log`: shrink the log file and write log data
10. `rvm_verbose`: control log verbosity

3 RVM DESIGN AND IMPLEMENTATION

Upon calling `rvm_init`, a directory is created if it does not exist along with a log file. Relevant segement list data structures are created as well. When a mapping is created with $\texttt{rvm}_{m}ap, a segment file is created if it do$

```
   Upon creating a transaction, a new transaction ID is created.  This transaction
ID is checked to prevent multiple transactions from editing the same segment.
Calling rvm_about_to_modify creates a range record.  Upon commit, the ranges
are actually written out to the log file.  Finally, in truncate, we read each
item from the log file and then apply the changes and write back the data to
disk.
```

3.1 RVM DATA STRUCTURES