15441 Paper Summaries, November 29, 2001
Suman Kumar Nath(sknath@cs.cmu.edu)

Recovery Management in QuickSilver

One line summary: Use atomic transaction as a mechanism to do failure recovery in the QuickSilver client-server structured system.

Main points:

- *Motivation* : Existing recovery techniques have problems. Example: *time-outs* : difficult to do correctly, consistency is difficult, *virtual circuits* : difficult to provide atomicity, *replication* : expensive.
- *QuickSilver solution* : Use atomic transaction to do failure recovery. The basic idea is as follows: clients and servers interact using IPC messages. Every IPC message belongs to a uniquely identified transaction, and is tagged with its transaction ID (TID). Servers tag the state they maintain on behalf of a transaction with its TID. IPC keeps track of all servers receiving messages belonging to a transaction, so that the transaction manager (TM) can include them in the commit protocol. TM's commit protocol is driven by calls from the client and servers, and by failure notifications from the kernel.Servers use the commit protocols messages as a signaling mechanism to inform them of failures, and as a synchronization mechanism for achieving atomicity. Recoverable servers call the Log manager (LM) to store their recovery data.
- *Architecture* : There is one recovery manager per host, and has three components: transaction manager (keeps track of the transactions the processes on the host participate in. Also manages commit protocols for the distributed hosts), log manager (common recovery log for TM commit's log and the server's data recovery data), and deadlock detector.
- *Transaction termination* : A TM terminates and initiates commit processing if the owner calls commit, if any participant calls abort, or if there is some sort of failure (possibly another node or a connection).
- *Committing* : Commit processing is similar to the standard two phase commit protocol, except that vote requests are propagated through the transaction's directed graph instead of having the coordinator communicate with all of the participants directly.
- *Recovery* : When a node crashes and is rebooted, LM scans the log starting at TM's log tail and determines status of each transaction known to TM at the time of crash. Pointers to the oldest and newest records for each transaction are stored, allowing servers to read only the blocks that contain their data. Finally, LM starts accepting identify requests.