DESIGN DOCUMENT

CDLList Class

This class is used to implement the data structure (as given).

Modifications:

1. Added variables in Element class:

* Public volatile Boolean iflock : provides lock on an Element node (true or false)
* Public volatile int count : keeps count of readers on an Element node

1. Added method in Write class :

* Public Cursor getCursor : returns Cursor

Class CDLCoarse

Functionality implemented by synchronizing methods with object of class Object.

Object lock=new Object();

synchronized(lock) { //on every method

<method definition> }

Class LockDefinition

Contains two Classes: 1) class FineGrain : functionality for fine grain locking

2) class ReadWriteLock : functionality for RW locks on CoarseGrain & FineGrain

Locking.

*Class FineGrain*

Defines two methods : public Boolean lock()

public Boolean unlock()

*Class ReadWriteLock*

defines 4 methods (according to API).

lockRead() & unlockRead() – for Coarse Grain R/W lock

lockWrite() & unlockWrite() – for Fine Grain R/W Lock

Class CDLListFine

Initialised FineGrain class and used its object to implement fine grain locking in “insert methods”.

Class CDLCoarseRW

Initialized class ReadWriteLock and implements CoarseGrain R/W lock with object of class ReadWriteLock.

CDLListFineRW

Initialized class ReadWriteLock and implements FineGrain R/W lock with object of class ReadWriteLock.

ALGORITHMS

1. **insertBefore() FineGrain lock (correspondingly for insertAfter)**

i)lock previous element

(if acquired, proceed else return false)

ii)Check if(previous==current) //only a single element in list

(if true)

iii)Insert new element -> unlock previous

else

iv)Lock current element

(if lock acquired, than)

v)Insert new element -> unlock previous and current

else

vi) unlock previous element -> exit

1. **insertBefore() FineGrainRW Lock (correspondingly for insertAfter)**

<follow same strategy as FineGrain lock, except acquiring RW lock>

1. **previous() FineGrainRW lock (correspondingly for next() )**

i)readlock current element

(if acquired)

ii)readlock previous element

(if acquired than)

iii)invoke super class method

else

iv)unlock current element