| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [Common/src](http://docs.google.com/index.html) - RetryAgent.cpp (source / [functions](http://docs.google.com/RetryAgent.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 5 | 39 | 12.8 % | | Date: | 2012-03-23 |  | Functions: | 4 | 8 | 50.0 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 0 | 12 | 0.0 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : :   2 : : #include "RetryAgent.h"  3 : : #include "RetryManager.h"  4 : : #include <unistd.h>  5 : : #include "time.h"  6 : : #include "print.h"  7 : :   8 : : namespace nft {  9 : :   10 : : //Constructor for this class  11 : :   12 : : RetryAgent::RetryAgent(const RetryManager \* inManager)   13 : 158207 : :mElapsedWaitTime(0),  14 : : mCurrentRetryCount(0),  15 : 158207 : mTimeToAdd(0) {  16 : 158207 : mManager = inManager;  17 : 158207 : mTimeToAdd = mManager->getRetryTimeInterval();   18 : : }  19 : :   20 : : //This function checks whether the agent has already waited for the specified number of times  21 : : //If it has, the function reports TIMEOUT  22 : : //Otherwise it returns AGAIN  23 : : RetryStatus  24 : 0 : RetryAgent::waitForSpecifiedTime() {  25 : 0 : uint32\_t retryInterval = mManager->getRetryTimeInterval();  26 : 0 : uint32\_t maxRetryTime = mManager->getMaxRetryTime();  27 : 0 : uint32\_t maxRetryCount = mManager->getTotalRetryCount();  28 : 0 : e9k\_debug("retryInterval:%d, maxRetryTime:%d,maxRetryCount:%d\n",  29 : : retryInterval,maxRetryTime,maxRetryCount);  30 : 0 : uint32\_t waitTime = 0;  31 [ # # ]: 0 : if (mCurrentRetryCount >= maxRetryCount) {  32 [ # # ]: 0 : if(mElapsedWaitTime >= maxRetryTime) {  33 : 0 : return eTimeOut;  34 : : }  35 [ # # ]: 0 : if ( maxRetryCount == 0 ) {  36 : 0 : waitTime = mTimeToAdd;  37 : : }  38 : 0 : mTimeToAdd = mTimeToAdd << 1;  39 [ # # ]: 0 : if (maxRetryCount != 0) {  40 : 0 : waitTime = mTimeToAdd;  41 : : }  42 : :   43 : : } else {  44 [ # # ]: 0 : if (mElapsedWaitTime >= maxRetryTime) {  45 : 0 : return eTimeOut;  46 : : }   47 : 0 : mCurrentRetryCount += 1;  48 : 0 : waitTime = retryInterval;  49 : : }   50 : 0 : timespec timeToWait = {  51 : : waitTime / 1000, // Seconds  52 : : (waitTime % 1000)\*1000000L // Nanoseconds  53 : 0 : };  54 : :   55 : 0 : timespec remainingTime = {0, 0};  56 : 0 : nanosleep(&timeToWait,&remainingTime);  57 : :   58 : 0 : mElapsedWaitTime += waitTime - remainingTime.tv\_sec/1000 - remainingTime.tv\_nsec\*1000000;   59 : 0 : e9k\_debug("mElapsedWaitTime:%d\n",mElapsedWaitTime);  60 : 0 : return eAgain;  61 : : }  62 : :   63 : :   64 : 158207 : RetryAgent::~RetryAgent(){  65 : :   66 : : }  67 : :   68 : :   69 : : RetryAgent::RetryAgent(const RetryAgent & source)   70 : 0 : :mElapsedWaitTime(source.mElapsedWaitTime),  71 : : mCurrentRetryCount(source.mCurrentRetryCount),  72 : : mTimeToAdd(source.mTimeToAdd),  73 : : mManager(source.mManager)   74 : 0 : {  75 : :   76 : : }  77 : :   78 : : RetryAgent &  79 : 0 : RetryAgent::operator =(const RetryAgent & source) {  80 [ # # ]: 0 : if(this != &source)  81 : : {  82 : 0 : mElapsedWaitTime = source.mElapsedWaitTime;  83 : 0 : mCurrentRetryCount = source.mCurrentRetryCount;  84 : 0 : mTimeToAdd = source.mTimeToAdd;  85 : 0 : mManager = source.mManager;  86 : : }  87 : 0 : return \*this;  88 : : }  89 : :   90 : :   91 : : } // namespace nft |

|  |
| --- |
| Generated by: [LCOV version 1.9](http://ltp.sourceforge.net/coverage/lcov.php) |