LogItem Class

1. int getDatacenterId()

This method gets data-center id. //in the cloud

1. void setDatacenterId(int datacenterId)

It sets the data-center id.

1. int getHostId()

It gets the hostid

1. void setHostId(int hostId)

It sets the hostid

1. int getVmId()

It gets the Id for VM

1. void setVmId(int vmId)

It sets the Id for VM

1. int getTaskLength()

It gets the task length

1. void setTaskLength(int taskLength)

It sets the task length

1. int getTaskInputType()

It gets the task input type

1. void setTaskInputType(int taskInputType)

It sets the task input type

1. int getTaskOutputSize()

It gets task output size

1. void setTaskOutputSize(int taskOutputSize)

It sets task output size

1. int getNumberOfHops()

It gets the nbr of hops (nbr of fog nodes data passes thru to reach its destination)

1. void setNumberOfHops(int numberOfHops)

It sets the nbr of hops

1. double getTaskStartTime()

It gets the task start time

1. void setTaskStartTime(double taskStartTime)

It sets the task start time

1. double getTaskEndTime()

It gets the task end time

1. void setTaskEndTime(double taskEndTime)

It sets the task end time

1. double getBwCost()

It gets the BW cost // BandWidth or BufferWriter

1. void setBwCost(double bwCost)

It sets the BW cost

1. double getCpuCost()

It gets the CPU cost

1. void setCpuCost(double cpuCost)

It sets the CPU cost

1. double getTaskCost()

It gets the cost of executing the task

1. void setTaskCost(double taskCost)

It sets the cost of executing the task

1. double getDistanceToHost()

It gets the distance from node to the host

1. double getDistanceToUser()

It gets the distance from node to the user

1. void setDistanceToHost(double distanceToHost)

It sets the distance from node to the host

1. void setDistanceToUser(double distanceToUser)

It sets the distance from node to the user

1. void setStatus(SimLogger.TASK\_STATUS status)

It sets the 13 task completion status given below –

1. CREATED,
2. UPLOADING,
3. PROCESSING,
4. DOWNLOADING,
5. COMPLETED,
6. REJECTED\_DUE\_TO\_VM\_CAPACITY,
7. REJECTED\_DUE\_TO\_BANDWIDTH,
8. UNFINISHED\_DUE\_TO\_BANDWIDTH,
9. UNFINISHED\_DUE\_TO\_MOBILITY,
10. ASSIGNED\_HOST,
11. REJECTED\_DUE\_TO\_LACK\_OF\_NODE\_CAPACITY,
12. REJECTED\_DUE\_TO\_LACK\_OF\_NETWORK\_BANDWIDTH,
13. REJECTED\_DUE\_TO\_UNACCEPTABLE\_LATENCY

1. void setVmType(int vmType)

It sets the VM type -- CLOUD\_VM

1. void setTaskType(int taskType)

It sets Task type.

1. void setNetworkDelay(double networkDelay)

It sets network delay

1. void setInWarmUpPeriod(boolean isInWarmUpPeriod)

It sets the InWarmUpPeriod based on T/F value.

1. void taskUploadStarted(double taskUploadTime)

It increases the network delay with taskUploadTime and makes task\_status = “UPLOADING”

1. void taskUploaded(int \_datacenterId, int \_hostId, int \_vmId, int \_vmType)

The task gets uploaded with datacenterid, hosted, vmid & vmType along with task\_status = “PROCESSING”

1. void taskDownloadStarted(double taskDownloadTime)

It increases the network delay with taskDownloadTime and makes task\_status = “DOWNLOADING”

1. void taskDownloaded(double \_taskEndTime)

It sets the task end time and makes task\_status = “COMPLETED”

1. void taskDownloaded(double \_taskEndTime, double cost)

It overloads #37 above with one extra parm = cost. It also sets the task end time and makes task\_status = “COMPLETED”

1. void taskRejectedStatus(double \_taskRejectTime, SimLogger.TASK\_STATUS taskStatus)

It sets the task end time and updates task\_status

1. void taskRejectedDueToVMCapacity(double \_taskRejectTime)

It sets the task end time and makes task\_status = “REJECTED\_DUE\_TO\_VM\_CAPACITY”

1. void taskRejectedDueToBandwidth(double \_taskRejectTime, int \_vmType)

It sets the task end time, vmType and makes task\_status = “REJECTED\_DUE\_TO\_BANDWIDTH”

1. void taskFailedDueToBandwidth(double \_time)

It sets the task end time and makes task\_status = “UNFINISHED\_DUE\_TO\_BANDWIDTH”

1. void taskFailedDueToMobility(double \_time)

It sets the task end time and makes task\_status = “UNFINISHED\_DUE\_TO\_MOBILITY”

1. void setCost(double \_bwCost, double \_cpuCos)

It sets the task execution cost with cpuCost and bwCost. //Bandwidth Cost

1. boolean isInWarmUpPeriod()

It sets a Boolean flag based on the task being in WarmUpPeriod or not.

1. double getCost()

It gets the task execution cost with cpuCost and bwCost

1. double getNetworkDelay()

It gets the Network Delay due to upload/download.

1. double getServiceTime()

It gets the service time as taskEndTime - taskStartTime

1. TASK\_STATUS getStatus()

It gets the 13 task completion status given in #29 above

1. int getVmType()

It gets the VM type as int. Allocates int nbr to CLOUD\_VM

1. int getTaskType()

It gets Task type as int.

1. String toString(int taskId)

Based on each task, it constructs a result string with – taskId, hostId, vmId, vmType, taskType, taskLength, taskInputType, taskOutputSize, taskStartTime & taskendTime separated by delimiter.

Next, it checks the task status.

If status = COMPLETED, result += networkDelay;

If status = REJECTED\_DUE\_TO\_VM\_CAPACITY, result += "1"; // failure reason 1

If status = REJECTED\_DUE\_TO\_BANDWIDTH, result += "2"; // failure reason 2

If status = UNFINISHED\_DUE\_TO\_BANDWIDTH, result += "3"; // failure reason 3

if status = UNFINISHED\_DUE\_TO\_MOBILITY, result += "4"; // failure reason 4

if status = REJECTED\_DUE\_TO\_LACK\_OF\_NODE\_CAPACITY, result += "5"; // failure reason 5

if status = REJECTED\_DUE\_TO\_LACK\_OF\_NETWORK\_BANDWIDTH, result += "6"; // failure reason 6

If status = REJECTED\_DUE\_TO\_UNACCEPTABLE\_LATENCY, result += "7"; // failure reason 7

Else result += "0"; // default failure reason

1. double getHostDist()

It gets the distance to the host

1. void setDistance(double in)

It sets the distance to the host

1. int getHops()

It gets the nbr of hops (nbr of fog nodes data passes thru to reach its destination)

1. int getHopsToUser()

It gets the nbr of hops data makes to reach the user

1. void setHops(int in)

It sets nbr of hops

1. void setHopsToUser(int in)

It sets the nbr of hops data makes to reach the user