# Data Structures

## Workload

The workload was stored as a queue of *Requests* in the *Workload* class. After parsing the *workload.txt* input file, a queue of *Requests* is created and ordered by the requested time. Each *Request* contains

* Source node name
* Destination node name
* The Path between the nodes (a list of nodes between source and destination)
* Timestamp and Duration

The path remains uninitialized until the request is processed and a path is calculated using the routing processor

## Network Topology

The network topology was stored as a graph in the *Network* class. The graph was broken into vertices for each node defined in *topology.txt* and Edges for each connection between nodes defined in *topology.txt*. Vertices were stored in a hash table referenced by their node-name stored as strings. Each vertex then contained a hash table of connected Edges, referenced by the other end of the Edge. Edges contain information on their propagation delay, virtual circuit capacity and number of active connections. The network also contains a queue of active connection requests, used to keep track of loading on the network. This data structure is illustrated in Figure 1.

Class *Network*

HashMap<String, Vertex> *Nodes*

“A” “B” “C”

Class *Vertex*

String *name* = “A”

HashMap<String, Edge> *adjacent*

“B”

“C”

Class *Vertex*

String *name* = “B”

HashMap<String, Edge> *adjacent*

“A” “C”

Class *Vertex*

String *name* = “C”

HashMap<String, Edge> *adjacent*

“B”

“A”

Class *Edge*

Class *Edge*

Class *Edge*

PriorityQueue <Request>

*activeConnections*

Figure - Diagram of network topology data structure