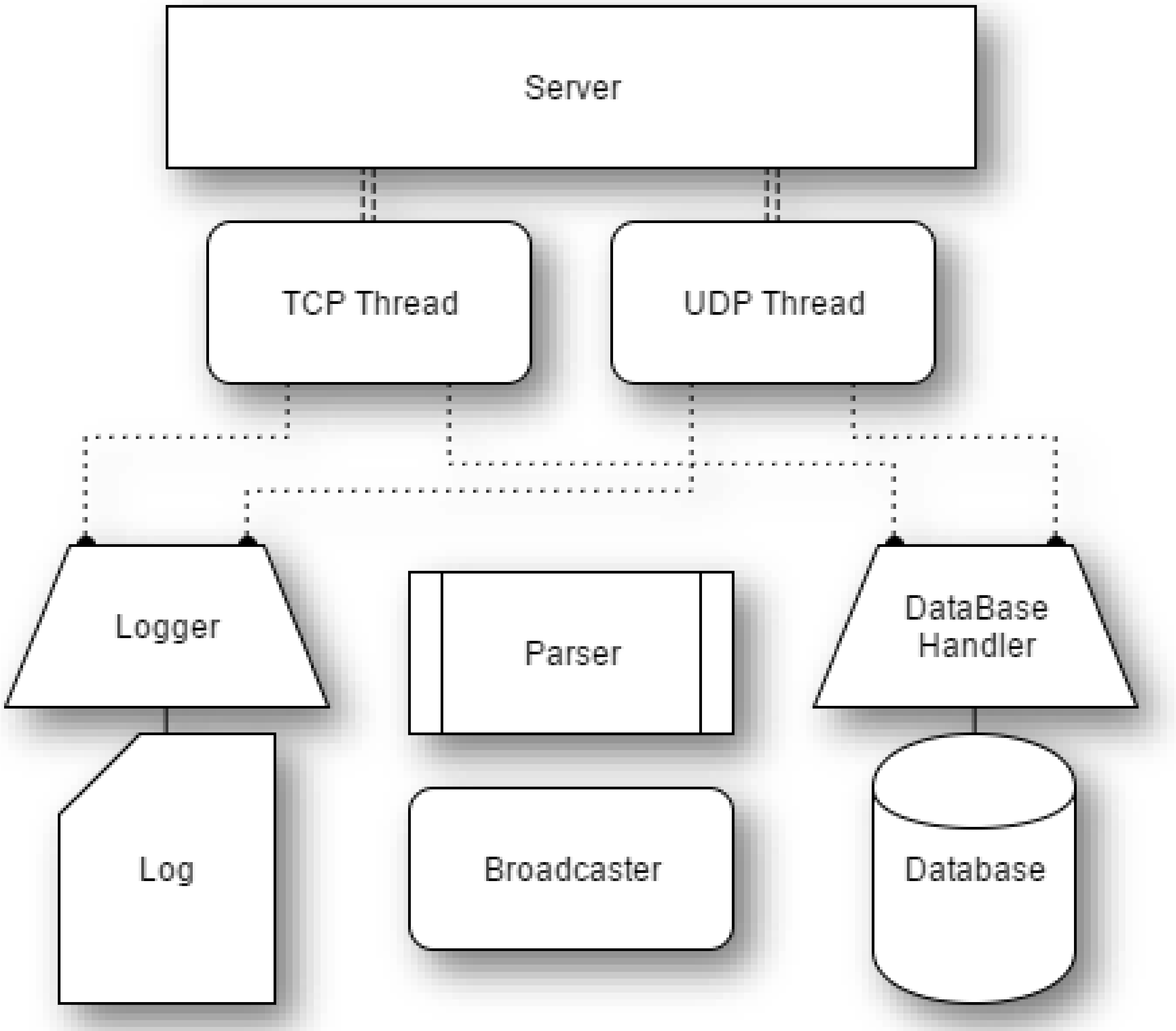
Architecture



- Server
  - public static void main(String[] args)
    - args -> command line arguments
    - Logger.Initialize()
    - GetOpt
    - DatabaseHandler.Initialize()
    - db.recreate()
    - TCPServer Thread
    - UDPServer Thread

- TCPServer
  - public void run()
    - ServerSocket
    - ClientSocket
    - PrintWriter
    - BufferedReader
    - out.println(Parser.Greetings)
    - String request
    - String response
    - Parser.ParseAndExecuteCommand(request, db)

- UDPServer
  - public void run()
    - DatagramSocket
    - Datagram Packet request
    - serverSocket.receive(request)
    - Parser.ParseAndExecuteCommand(request, db)
    - Datagram Packet response
    - serverSocket.send(response)

- Parser
  - private static String ParseSelectedPeers(List<String[]> peers)
    - peers -> known peers from database
    - for each peer in peers: string.append(peer)
    - return peerResponse.toString()
  - static String ParseAndExecuteCommand(String request, DatabaseHandler db)
    - request -> client command + parameters
    - db -> handle to the sqlite database connection
    - request.split(Parser.inSep)
    - switch(command) { “GOSSIP”, “PEER”, “PEERS?” }
    - db.exists(sha)
    - db.insertGossip(sha, dt, message)
    - Broadcaster.getInstance().broadcast(request, db.selectPeers())
    - db.insertPeer(name, port, ip)

- Broadcaster
  - Static Broadcaster getInstance()
  - void broadcast(String message, List<String[]> peers)
    - message -> successfully inserted gossip request
    - peers -> known peers from database
    - for each peer in peers: clientSocket.send(new DatagramPacket(gossip, ip, port))

- Logger
  - static Logger Initialize(String path, boolean append, boolean debug)
    - path -> path to log file
    - append -> append mode or create mode
    - debug -> debug mode or production mode
  - static Logger getInstance()
  - public void close()
  - void log(Exception ex)
  - void log(String str)
  - void log(String type, String side, String str)
    - write("<timestamp>: [tcp/udp][server/client] <string>")

- DataBaseHandler
  - static DataBaseHandler Initialize(String connectionString)
    - connectionString -> path to database file
  - static DataBaseHandler getInstance()
  - public void close()
  - void recreate()
    - DROP Peer
    - DROP Gossip
    - CREATE Peer
    - CREATE Gossip
  - void insertPeer(String name, String port, String ip)
    - UPSERT Peer
  - void insertGossip(String sha, String dt, String message)
    - dt -> datetime portion of request
  - boolean exists(String sha)
    - sha -> gossip sha value and PK for Gossip
  - List<String[]> selectPeers()
    - ResultSet