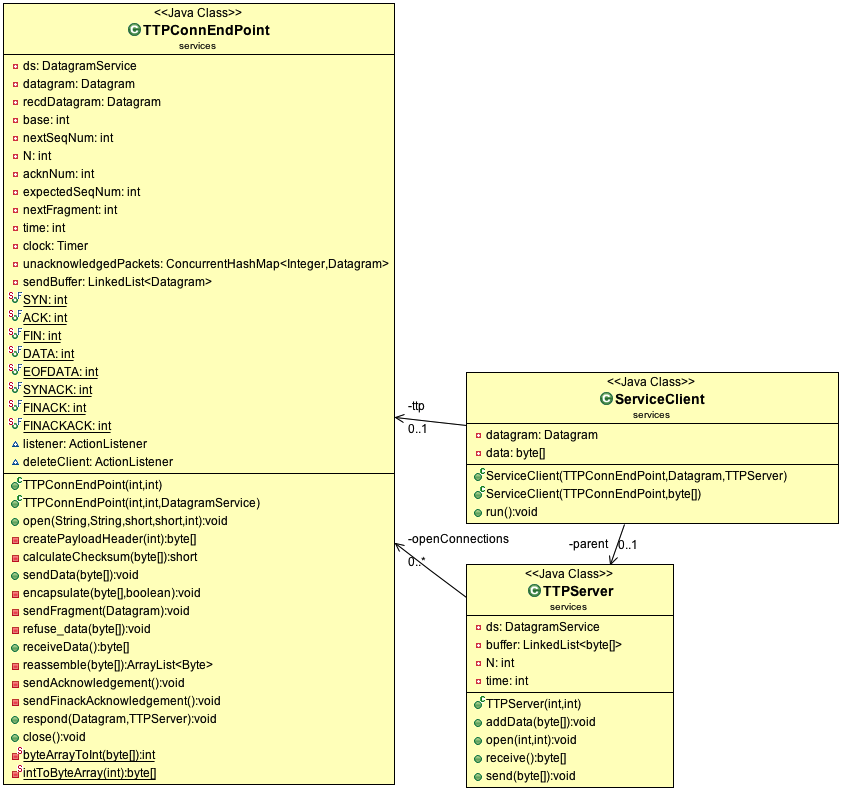
**CLASS DIAGRAM:**



**NOTE:**

TTPConnEndPoint class, representing an end point of a reliable connection, can be considered analogous to a Socket. It contains the entire logic of Go-Back-N mechanism.

The TTPServer listens for incoming SYN connection requests from clients (running TTPConnEndPoint) and creates a new instance of TTPConnEndPoint on the server side to handle each connection. The TTPServer thereafter continues listening for new connection requests and the actual work of interfacing with the client endpoint is handled by the server TTPConnEndPoint.

Program Flow:

1. The FTPServer is configured using Run Configuration to accept 2 command line arguments i.e. TTPServer send window size and TTPServer retransmission timer interval. FTPServer instantiates TTPServer using these parameters.
2. It calls open(int port, int verbose) on TTPServer instance which in turn initializes a DatagramService instance on the specified port.

Note: FTP Server is hardcoded to listen on port 2221 in this case for testing purposes since FTP port 21 is already bound.

1. FTPServer listens continuously for file requests on Port 2221 by calling receive() on TTPServer instance in a while(true) loop. TTPServer receive() in turn calls receiveDatagram() of the underlying DatagramService instance.
2. **FTPClient** is configured using Run Configuration in Eclipse to accept 2 command line arguments i.e. send window size and retransmission timer interval.
3. It creates an instance of TTPConnEndPoint, which contains the entire logic for Go-Back-N mechanism. TTPConnEndPoint class, representing an end point of a reliable connection, can be considered analogous to a Socket.
4. It calls open(int port, int verbose) on this instance which in turn initializes a DatagramService instance on the specified port.