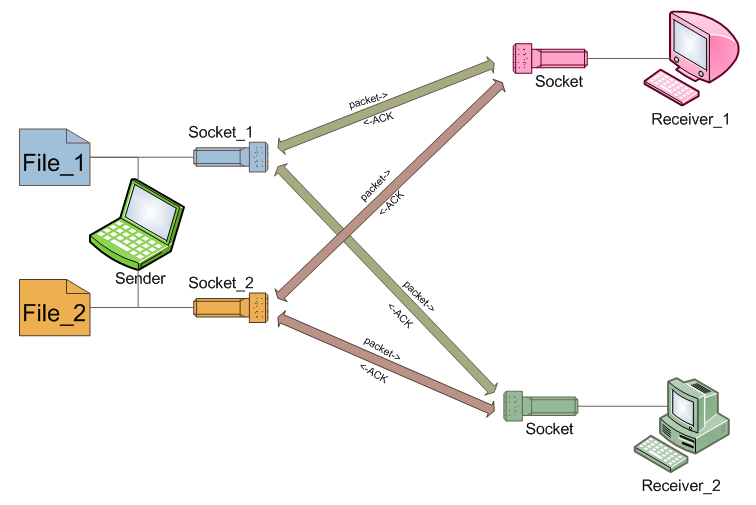
# The Idea



The sender can transfer multiple files to several receivers, as illustrated in the figure above, there are two files on the sender’s side, and each file binds to a socket. Since there two receivers, a file is sent via its associated socket to all the receivers respectively.

# The Data Structure

We define several data structures in ***rudp\_api.h***

// define the states of a (file transfer) session

**typedef** **enum** **{**

RSESSION\_START**,** // session has started

RSESSION\_TRANSFER**,** // transfering

RSESSION\_CLOSED**,** // session has been closed

WAIT\_FOR\_ACK\_OF\_FIN // waiting for last ACK (for FIN)

**}** enum\_session**;**

// a datagram encapsulates the data to be sent/received

**struct** r\_datagram **{**

**struct** rudp\_hdr header**;** // type of data

**char** data**[**RUDP\_MAXPKTSIZE**];** // data itself

**int** len**;** // length of data

**int** has\_ack**;** // whether this data has been ACKed or not

**int** has\_send**;** // whether this data has been sent or not

**int** retrans\_num**;** // number of retransmission times

**struct** r\_datagram**\*** next**;** // pointer to next datagram

**struct** sockaddr\_in remote\_addr**;** // remote socket

**struct** r\_socket**\*** rsocket**;** // local socket

**struct** r\_database**\*** database**;** // pointer to the database

**};**

// a database is used to maintain different sessions

**struct** r\_database **{**

**struct** sockaddr\_in **\***remote**;** // remote socket

**int** is\_initialed**;** // -1 = false; 1 = true;

**int** last\_recv\_seq**;** // last recived sequence number

**int** last\_send\_seq**;** // last sent sequence number

enum\_session session\_state**;** // holds the state of session

**struct** r\_datagram**\*** datagram\_buffer**;** // holds datagrams

**struct** r\_database**\*** next**;** // pointer to next database

**struct** r\_database**\*** pre**;** // pointer to previous database

**};**

**typedef** **struct** r\_database**\*** r\_database\_t**;**

// encapsulates a socket

**struct** r\_socket **{**

**int** **(\***super\_rudp\_receiver**)(struct** r\_socket**\*,** **struct** sockaddr\_in**\*,** **char\*,** **int);**

**int** **(\***super\_event\_handler**)(struct** r\_socket**\*,** rudp\_event\_t**,** **struct** sockaddr\_in **\*);**

**int** sd**;** // socket descriptor

// used by recevier to filter incoming file (identified by remote\_socket and seq)

**struct** r\_database**\*** database**;**

**};**

**typedef** **struct** r\_socket**\*** rudp\_socket\_t**;**

# State Machine

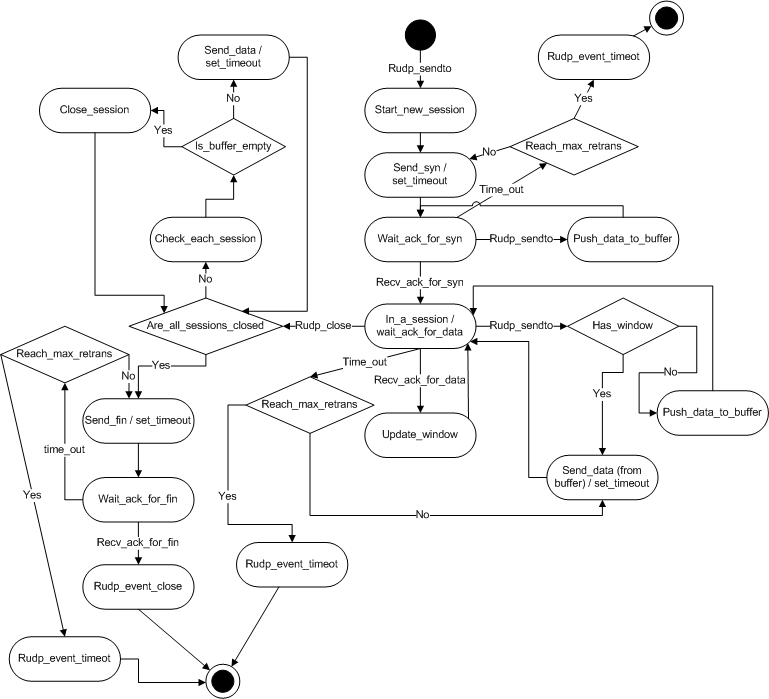
Sender

Figure Sender

Refer to **vs\_send\_statechart.png**

## Receiver

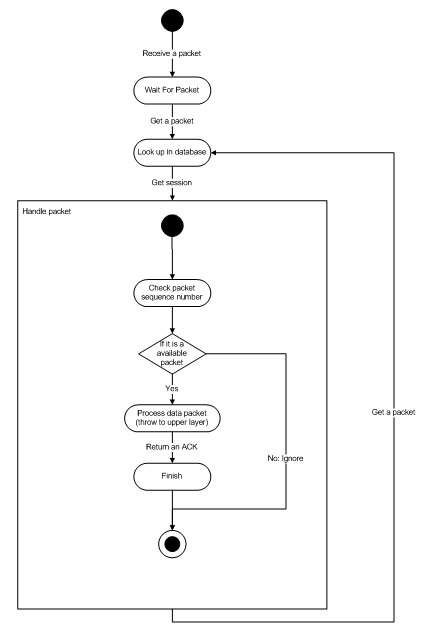


Figure Receiver

Refer to **vs\_recv\_statechart.png**