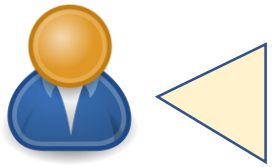
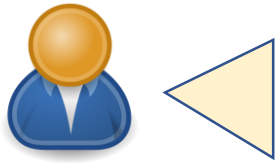
RELIABLE UDP FILE TRANSFER: RUFT

TYPE SRC ADDRESS - PORT (*) FILE NAME



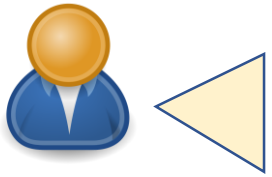
VIEW REQUEST

LIST IP1-PORT P



UPLOAD REQUEST

PUT IP2-PORT X PAPEROGA



DOWNLOAD REQUEST

GET IP3-PORT F FILENAME1

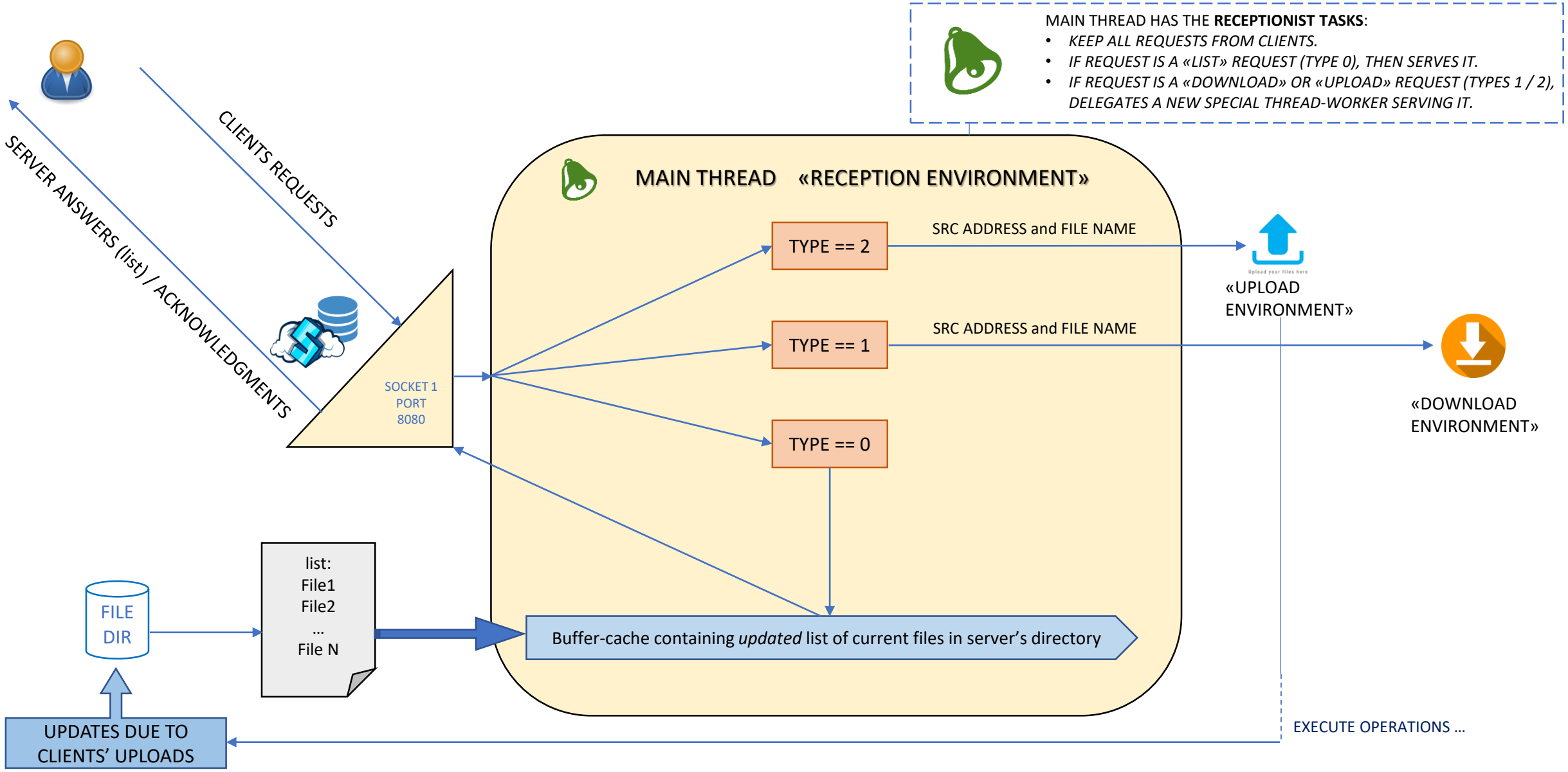
WELL KNOWN RECEPTION PORT RECEPTION SOCKET.

- SERVER'S INTERNAL STRUCTURE IS LOGICALLY DIVIDED INTO THREE COMMUNICATING PARTS, EACH ONE MATCHES A SPECIFIC REQUEST TYPE.

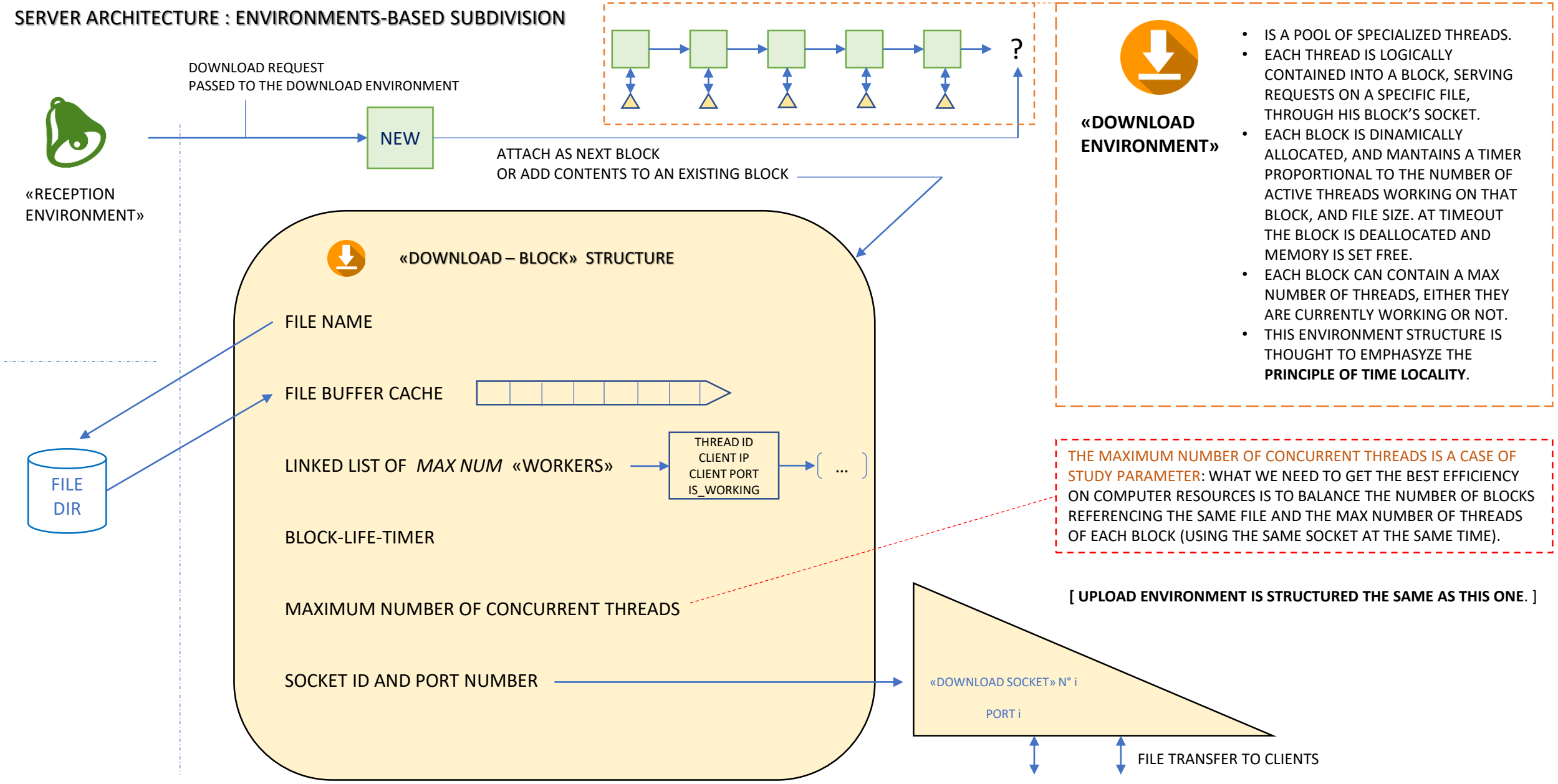
- OF COURSE SERVER'S INTERNAL STRUCTURE IS NOT VISIBLE TO THE CLIENT, WHO JUST COMMUNICATES WITH THE RECEIVING SERVER'S SOCKET.

- THIS SOFTWARE CONSISTS IN A CLIENT-SERVER UDP APPLICATION.
- RELIABLE DATA TRANSFER IS IMPLEMENTED WITHIN THE APPLICATION LAYER.
- CLIENTS CAN SEND VISUALIZATION/UPLOAD/DOWNLOAD REQUESTS TO SERVER .

SERVER ARCHITECTURE : ENVIRONMENTS-BASED SUBDIVISION



SERVER ARCHITECTURE : ENVIRONMENTS-BASED SUBDIVISION



RDT : SLIDING WINDOW PROTOCOL IMPLEMENTATION. SENDER

- THE **SEQUENCE NUMBER** REPRESENTS THE POSITION OCCUPIED BY A PACKET WITH RESPECT TO THE OTHERS, WITHIN THE FILE THAT'S BEING TRANSMITTED.
  SEQUENCE NUMBER IS DEFINED IN AN INTEGER RANGE :

  { 0 ; (FILESIZE (bytes) / PACKETSIZE (bytes)) }

- THE **SLIDING WINDOW** HAS A **SIZE** OF **N**:
  THAT MEANS AT MOST N PACKETS ARE ALLOWED TO BE ON-THE-FLY AT THE SAME TIME, WITHOUT ACKNOWLEDGMENT.

- GOING-TO-BE-SENT PACKETS CAN BE TEMPORARILY STORED INTO A **READY-QUEUE**, WAITING THE WINDOW TO BE FREE.

TIMER

- EACH PACKET HAS A TIMEOUT COUNTER. AS THE COUNTDOWN REACHES THE 0 AND NO ACKS HAVE BEEN DETECTED FOR THAT PACKET, THEN IT IS TO BE SENT AGAIN.

- *THE APPLICATION HAS TO SIMULATE THE LOSS OF PACKET WITH A PROBABILITY OF «P», RUNTIME-SET PARAMETER.*

RECEIVING ACKS

- WHEN AN ACKNOWLEDGMENT IS RECEIVED BY THE CLIENT SIDE, THE RESPECTIVE ENTRY WITHIN THE SLIDING WINDOW IS MARKED AS ACKED.

- IF THE ENTRY IS LOCATED AT THE BEGINNING OF THE SLIDING WINDOW, THEN THE WINDOW SLIDES UP.

STRUCT SLOT

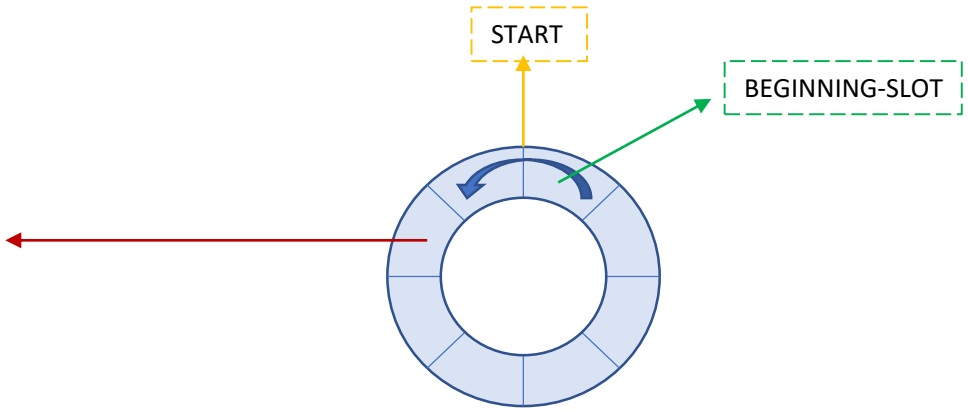
```
int     sequence_number;

int     status;

/* FREE = 0
   SENT = 1
   ACKED = 2   */

boolean   is_first;
```

CIRCULAR BUFFER (OF SIZE N) IMPLEMENTS THE SLIDING WINDOW STRUCTURE :

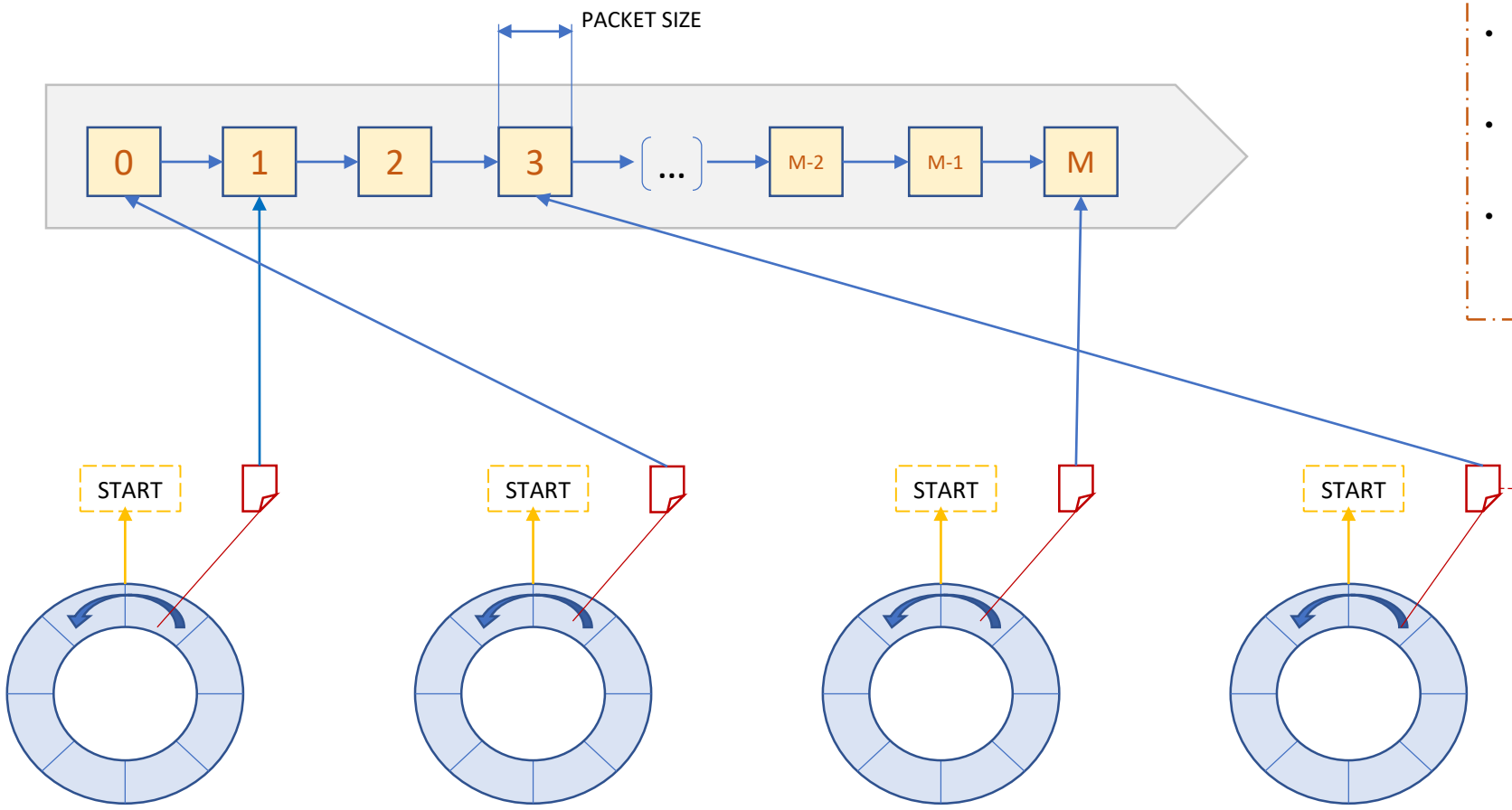
- EVERY TIME A NEW PACKET HAS TO BE SENT, IF THE BUFFER IS NOT «FULL», PACKET'S SEQUENCE NUMBER IS ASSIGNED TO THE FIRST FREE SLOT IN THE CIRCULAR BUFFER (FOLLOWING THE ORDER FROM THE START), AND THE PACKET IS SENT.

- AS THE CIRCULAR BUFFER GETS FILLED, IT WOULDN'T BE POSSIBLE TO TRANSMIT NEW PACKETS, UNTILL ONE OR MORE CONSECUTIVE BEGINNING-SLOTS ARE ACKNOWLEDGED BY THE SERVER SIDE.

- ONCE THIS EVENT OCCURS, THE «X» CONSECUTIVE ACKNOWLEDGED SLOTS ARE SET FREE, THUS THE BUFFER SLIDES LEFT OF «X», THE START INDEX IS INCREMENTED OF X % N, AND THE WINDOW HAS «X» FREE SLOTS TO BE USED BY NEW PACKETS TO BE SENT.



RDT : SLIDING WINDOW PROTOCOL IMPLEMENTATION. SENDER



WINDOW 1 : CLIENT 1

WINDOW 2 : CLIENT 2

WINDOW 3 : CLIENT 3

WINDOW 4 : CLIENT 4

DOWNLOAD-BLOCKS EXAMPLE

- EACH OF THE DOWNLOAD-BLOCKS REFERS TO A SPECIFIC FILE STORED IN SERVER'S DIRECTORY.
- THE FILE IS PUT INTO A STREAM (BUFFER CACHE) AND SHARED BY A NUMBER OF CLIENTS TRYING TO DOWNLOAD IT.
- FOR EACH CLIENT SHOULD BE POSSIBLE CREATING AD-HOC SLIDING WINDOW OPERATING ON SHARED PACKETS.
- EACH WINDOW IS OF COURSE MATCHED WITH THE BLOCK'S OUTPUT SOCKET, SO IT HAS TO BE SHARED TOO.

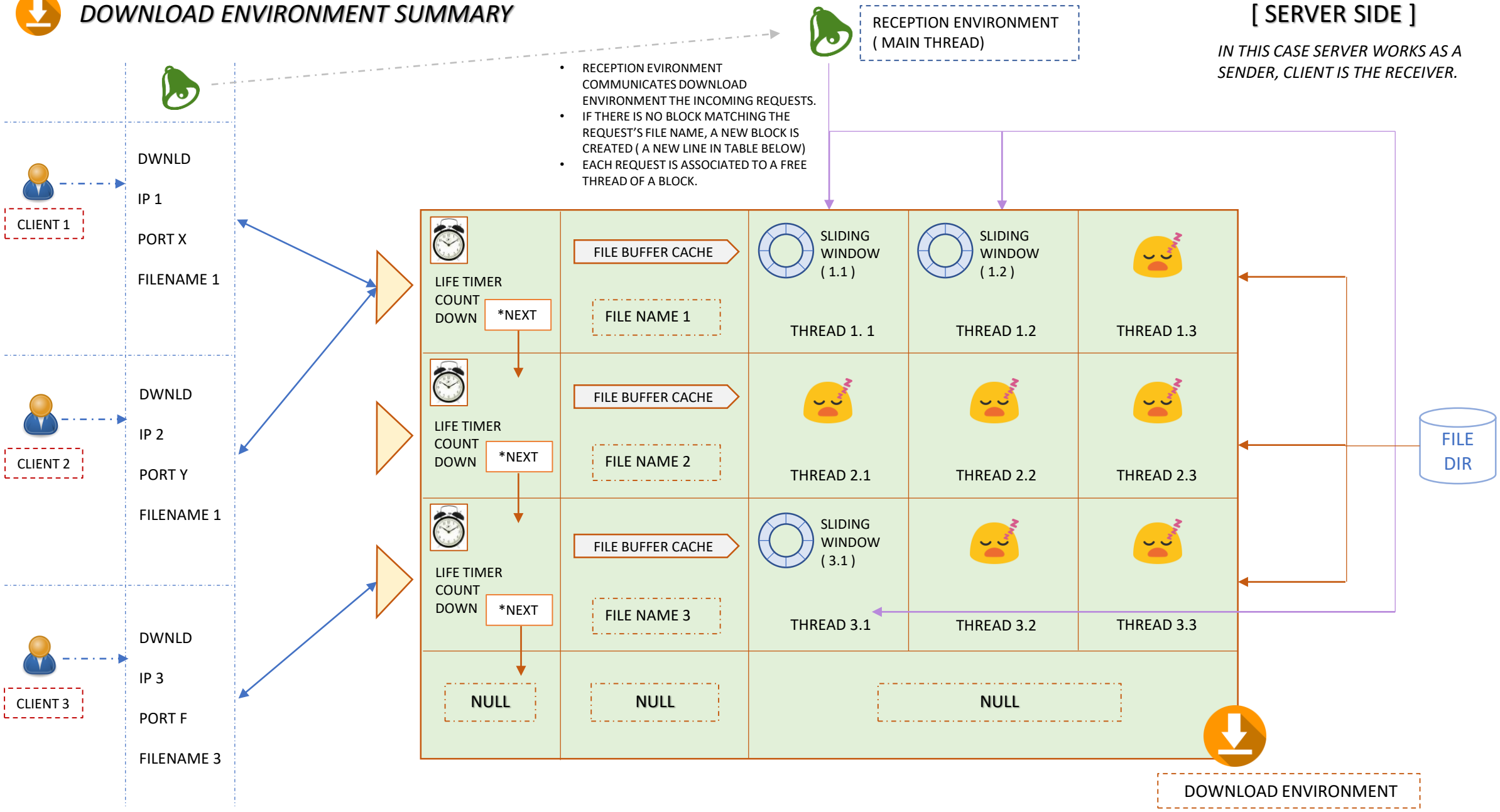
STRUCT SLOT

```
int     sequence_number;

int     status;

/* FREE = 0
   SENT = 1
   ACKED = 2  */

boolean   is_first;
```

DOWNLOAD ENVIRONMENT SUMMARY

[SERVER SIDE]

IN THIS CASE SERVER WORKS AS A SENDER, CLIENT IS THE RECEIVER.



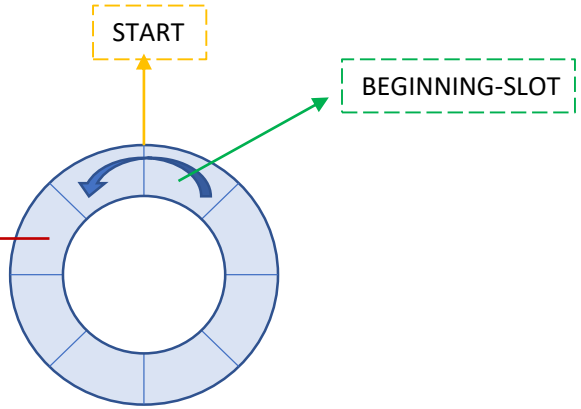
- RECEPTION EVIRONMENT COMMUNICATES DOWNLOAD ENVIRONMENT THE INCOMING REQUESTS.
- IF THERE IS NO BLOCK MATCHING THE REQUEST'S FILE NAME, A NEW BLOCK IS CREATED (A NEW LINE IN TABLE BELOW)
- EACH REQUEST IS ASSOCIATED TO A FREE THREAD OF A BLOCK.

RDT : SLIDING WINDOW PROTOCOL IMPLEMENTATION. RECEIVER

RECEIVER SIDE, SLIDING WINDOW PROTOCOL REQUIRES A RECEIVING WINDOW OF THE SAME SIZE OF SENDER'S SLIDING WINDOW. THUS, THE SAME (OR REALLY SIMILAR) STRUCTURE EXPLAINED FOR SENDER'S SIDE IS ADOPTED IN HERE.

STRUCT SLOT

```
int     sequence_number;

int     status;

/* WAITING = 0
   RECEIVED = 1 */

boolean   is_first;
```



AS A PACKET IS RECEIVED:

- IF ITS SEQUENCE NUMBER IS ONE CONTAINED WITHIN THE RECEIVING WINDOW, THEN RECEIVER REPLIES THE SENDER WITH A **SELECTIVE ACKNOWLEDGMENT**.
- IF ITS SEQUENCE NUMBER IS ONE CONTAINED WITHIN THE RECEIVING WINDOW BUT THE SLOT SIGNALS IT HAS ALREADY BEEN RECEIVED, YET THE RECEIVER REPLIES THE SENDER WITH A SELECTIVE ACKNOWLEDGMENT.
- IF ITS SEQUENCE NUMBER MATCHES THE BEGINNING-SLOT'S SEQUENCE NUMBER, THE RECEIVING WINDOW SLIDES LEFT OF A NUMBER OF POSITIONS EQUAL TO THE CONSECUTIVE SLOTS (FROM THE BEGINNING) HAVING STATUS EQUAL TO «RECEIVED», AND RECEIVING WINDOW IS UPDATED WITH NEW SEQUENCE NUMBERS.
- IF ITS SEQUENCE NUMBER IS NOT CONTAINED WITHIN THE RECEIVING WINDOW, THEN THE RECEIVER IGNORES THE PACKET.

CONCERNING TIMERS : *DOWNLOAD ENVIRONMENT*

BLTC : Block Life Timer Countdown

AS A NEW BLOCK (REFERENCING A FILE STREAM) IS ALLOCATED, A BLTC VARIABLE IS DECLEARED. THE DEFAULT VALUE OF THIS TIMER IS PROPORTIONAL TO THE NUMBER OF PACKETS REQUIRED TO TRANSFER THE FILE, THAT IS :

FILESIZE (BYTES) / PACKETSIZE (BYTES)

THE PROPORTIONALITY FACTOR HAS TO BE A FRACTION OF SECONDS, AND THIS SHOULD BE A CONFIGURABLE PARAMETRIC VALUE, NAMED «TAO».

FURTHER, BLTC IS DINAMICALLY INCREMENTED BY THE NUMBER OF TOTAL ACCESSES TO THE FILE (DUE TO CLIENTS' REQUESTS), SO THAT THE EQUATION WOULD FINALLY BE :

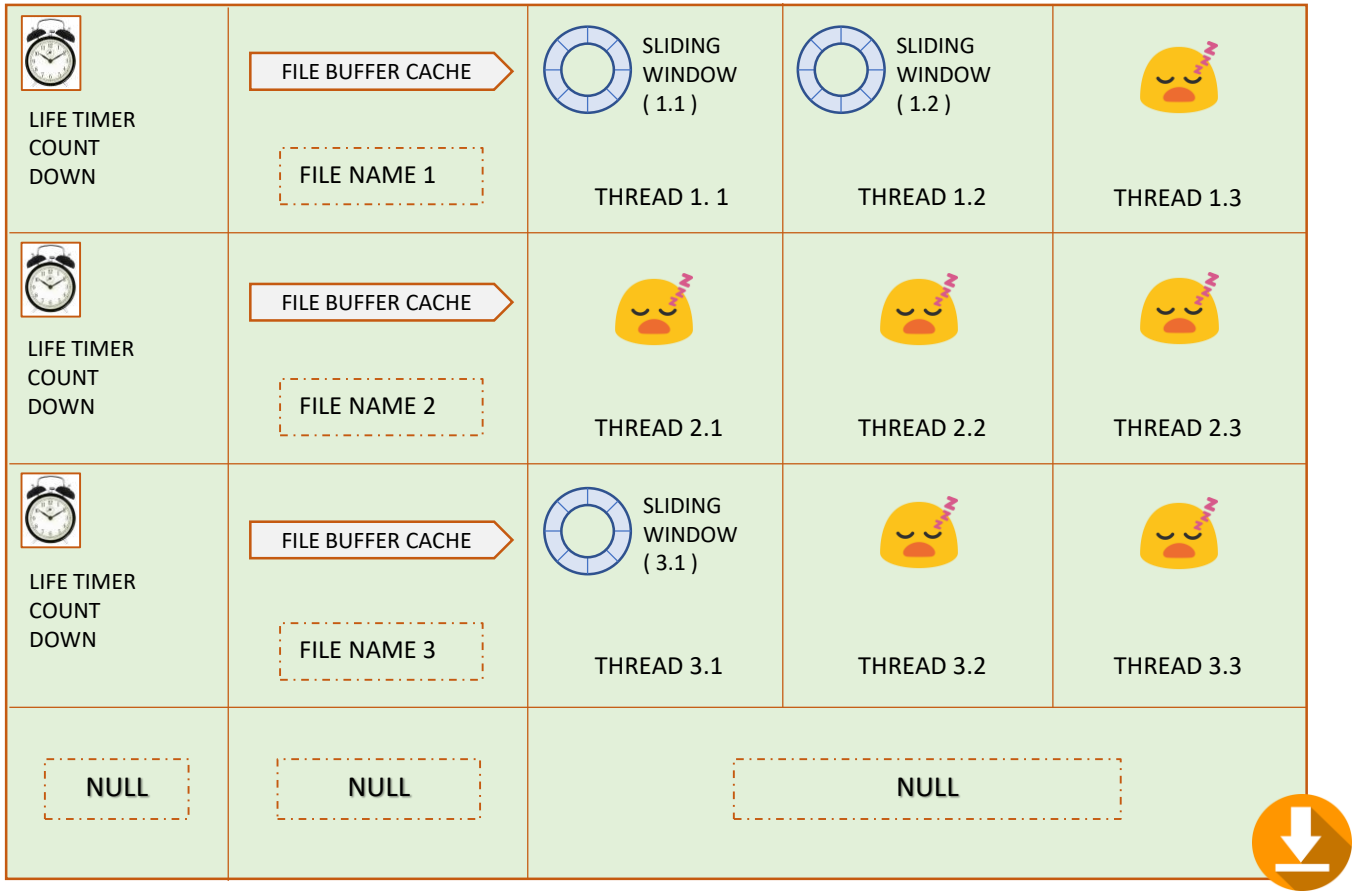
BLTC = (NUMPACKET + ACCESSES) tao

IN EXAMPLE, IF THE TRANSFERRED FILE SIZE IS 4 KB, AND PACKET SIZE IS 512 B, AND THE NUMBER OF TOTAL ACCESSES IS CURRENTLY 2 (AS THE FIRST LINE CASE ON THE TABLE ON THE SIDE), THEN THE BLTC VALUE WOULD BE :

BLTC = (4096 / 512) + 2 = 8 + 2 = 10 tao.

THE COUNTSDOWN ACTUALLY STARTS WHEN ALL THREADS OF THE BLOCK HAS FINISHED THEIR TASKS (AS THE SECOND LINE CASE ON THE TABLE ON THE SIDE) :

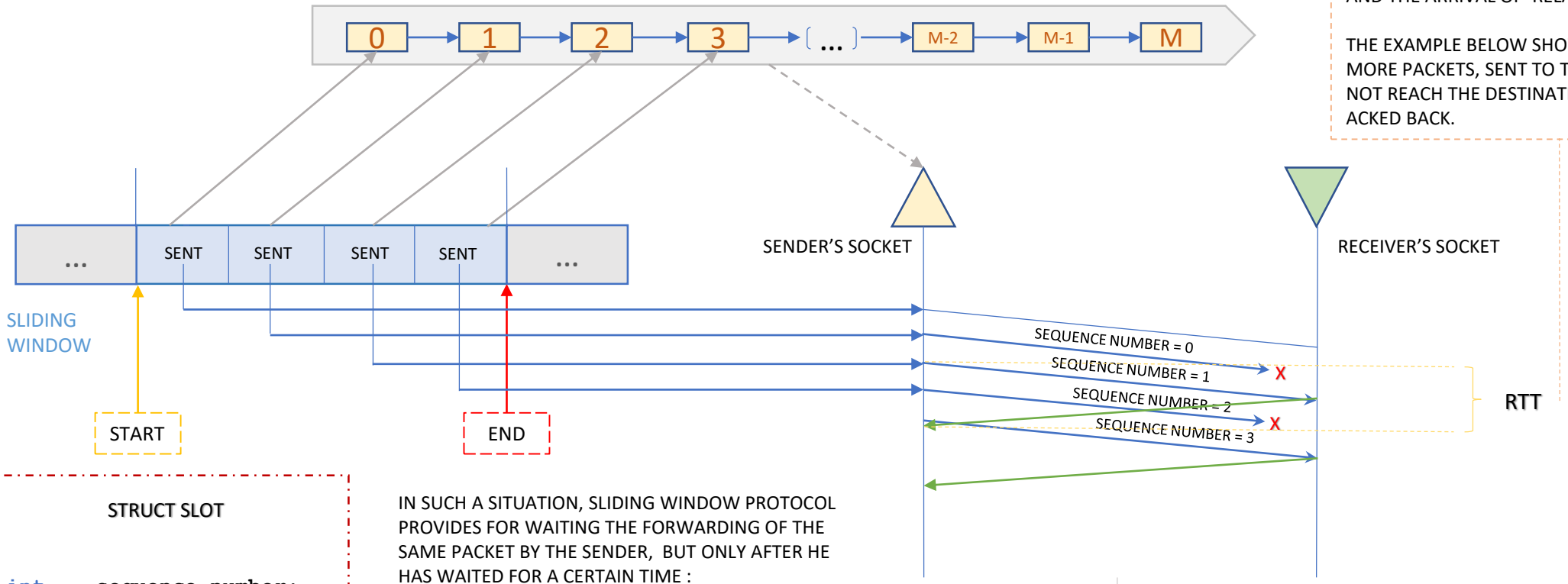
- IF THE TIMER RUNS OUT AND NO NEW REQUESTS HAS BEEN KEPT BY THE BLOCK, THE LATTER IS DEALLOCATED WITH ALL SUBSTRUCTURES AND WORKING THREADS.
- ELSE, IF A ONE OR MORE NEW REQUEST ARE KEPT BY THE BLOCK WHILE TIMER IS STILL RUNNING, THE TIMER IS RESET TO HIS LAST VALUE + NUMBER OF NEW ACCESSES. WHEN ALL OF THESE REQUESTS HAVE BEEN SERVED, TIMER STARTS AGAIN WITH ITS NEW VALUE.



CONCERNING TIMERS : *ESTIMATED RTT & ADAPTIVE TIMEOUT SETTINGS*

THE MEASURED ROUND TRIP TIME (RTT) IS DEFINED AS THE TIME BETWEEN THE FORWARDING OF THE PACKET AND THE ARRIVAL OF RELATED ACKNOWLEDGMENT.

THE EXAMPLE BELOW SHOWS THE CASE WHERE ONE OR MORE PACKETS, SENT TO THE RECEIVER, ACTUALLY DO NOT REACH THE DESTINATION, THEREFORE THEY ARE NOT ACKED BACK.



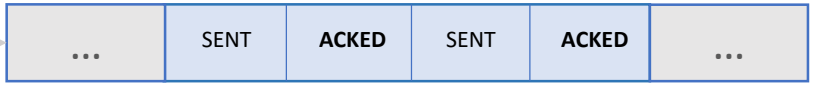
STRUCT SLOT

```
int      sequence_number;

int      status;

/* FREE = 0
   SENT = 1
   ACKED = 2  */

boolean    is_first;
```

IN SUCH A SITUATION, SLIDING WINDOW PROTOCOL PROVIDES FOR WAITING THE FORWARDING OF THE SAME PACKET BY THE SENDER, BUT ONLY AFTER HE HAS WAITED FOR A CERTAIN TIME :

- IF THE TIMER RUNS OUT TOO EARLY, THE RISK IS TO OPERATE A NOT NECESSARY RETRANSMISSION.
- IF THE TIMER RUNS OUT TOO LATE, IT TAKES TO A LOW-PERFORMANCE SCENARIO.

FOR THESE REASONS IS NECESSARY TO SET A TIMEOUT INTERVAL THAT LASTS A LITTLE MORE THAN THE ROUND TRIP TIME. THEREFORE, NEXT STEPS ARE *ESTIMATE RTT VALUE* AND *CALCULATE THE TIMEOUT INTERVAL*.



CONCERNING TIMERS : *ESTIMATED RTT & ADAPTIVE TIMEOUT SETTINGS*

THE FOLLOWING SOLUTION IS THE SAME ADOPTED BY TCP, BUT IS IMPLEMENTED ON THE APPLICATION LAYER.

STEP 1 : ESTIMATED RTT & DEV RTT

IN ORDER TO GET A REALISTIC VALUE OF ESTIMATED RTT, IT'S NECESSARY THE MEASURE OF SAMPLE RTT ON EACH PACKET **TRANSMITTED ONCE**.
THE ESTIMATED RTT VALUE IS GAINED BY A WEIGHTED AVERAGE ON PREVIOUS ESTIMATED RTT VALUE AND THE NEW MEASURED SAMPLE RTT VALUE:

*ESTIMATED RTT = (0.875 * ESTIMATED RTT) + (0.125 * SAMPLE RTT)*

A MAJOR WEIGHT IS GIVEN TO THE PREVIOUS VALUE OF ESTIMATED RTT, FOLLOWING THE PRINCIPLE OF TIME LOCALITY.

FURTHER, IT IS NECESSARY TO CONSIDER THE *VARIABILITY OF THE ROUND TRIP TIME*
(DEV RTT) : IT ESTIMATES THE DIFFERENCE BETWEEN THE NEW VALUE OF SAMPLE RTT AND THE ESTIMATED RTT VALUE.

*DEV RTT = (0.75 * DEV RTT) + (0,25 * (SAMPLE RTT – ESTIMATED RTT))*

STEP 2 : TIMEOUT INTERVAL

IT IS NOW POSSIBLE TO CALCULATE DINAMIC VALUES OF RETRANSMISSION TIMEOUT INTERVAL.
THE TRICK IS TO CALCULATE A TIMEOUT INTERVAL VALUE SUCH THAT IT DEVIATES A FEW FROM ESTIMATED RTT IF THERE'S NO TRAFFIC IN THE NET, BUT MUCH OTHERWISE.

*TIMEOUT INTERVAL = ESTIMATED RTT + (4 * DEV RTT)*

Reliable Data Transfer : IMPLEMENTATION ORIENTED BLOCK-STRUCTURE'S UPGRADE

IN ORDER TO IMPLEMENT THE RELIABLE DATA TRANSFER LOGIC, IT HAS BEEN NECESSARY THE IMPROVEMENT OF BLOCK'S ARCHITECTURE ITSELF.

ACKNOWLEDGMENT KEEPER THREAD

ACCORDING TO THE RDD APPROACH (RESPONSIBILITY DRIVEN DESIGN), EACH THREAD SHOULD HAVE A PRECISE TASK. THE ACKNOWLEDGMENT KEEPER THREAD IS AN ADDICTIVE THREAD FOR EVERY BLOCK IN DOWNLOAD ENVIRONMENT.

ITS TASK IS TO BE CONTINOUSLY RECEIVING DATA FROM BLOCK'S SOCKET : FOR SURE, EACH RECEIVED MESSAGE OF A DOWNLOAD ENVIRONMENT'S BLOCK IS AN ACK.

- THE KEEPER TAKES THE PACKET, UNDERSTANDS WHICH WORKER IT IS RELATED TO, AND UPDATES THAT WORKER'S SLIDING WINDOW'S SLOT STATUS TO «ACKED».
- FURTHER, IF THIS WINDOW'S SLOT IS THE FIRST OF THE SLIDING WINDOW, THE KEEPER FORWARD A SIGNAL TO THE SPECIFIC WORKER, WHO AWAKES FROM A PAUSE AND SLIDES THE WINDOW ON, AND GOES ON WITH THE TRANSFER.

Reliable Data Transfer : IMPLEMENTATION ORIENTED BLOCK-STRUCTURE'S UPGRADE

SEQ NUM

STATUS

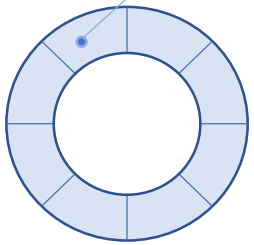
IS FIRST

CHAR * PACKET

TIMEOUT INTERVAL

SENT TIMESTAMP

ACKED TIMESTAMP



ACKNOWLEDGMENT KEEPER THREAD
(*one for each block*)

STEP 3 - AKNOWLEDGMENT KEEPER

- RECEIVES ACKNOWLEDGMENTS ADDRESSED TO EVERY WORKER OF THE BLOCK.
- ACCESSES THE REFERRED WORKER.
- ACCESSES THE REFERRED WINDOW'S SLOT, UPDATING THE ATTRIBUTE STATUS VALUE TO ACKED.
- IF THE SLOT IS THE FIRST OF THE WINDOW, SIGNALS THE WORKER TO SLIDE THE WINDOW ON AND CONTINUE WITH TRANSFER.

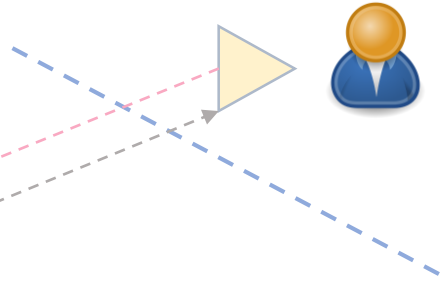
ACK (X, i)

PACKET (i)

WORKER THREAD «X»

STEP 1 - THE WORKER

- BUILDS THE PACKET (i) AND SETS IT AS ATTRIBUTE «PACKET» OF SLIDING WINDOW'S SLOT (i).
- SETS THE ATTRIBUTE «STATUS» OF WINDOW'S SLOT TO «SENT».
- SENDS THE PACKET (i) TO THE CLIENT.
- SETS THE ATTRIBUTE «SENT TIMESTAMP» OF WINDOW'S SLOT TO THE CURRENT TIMESTAMP .



STEP 2 - CLIENT

- RECEIVES FILE PACKETS AND NOTIFY SERVER WITH ACKNOWLEDGMENTS.

TIME WIZARD THREAD

STEP 1 - THE TIME WIZARD

WHILE WORKER THREAD IS RUNNING (TRANSMITTING PACKETS) THE TIME WIZARD THREAD REPEATS THE FOLLOWING CYCLE :

- NANO-SLEEPS (CLOCK TIMER)
- COMPARES THE CURRENT TIMESTAMP WITH THE SENDING-TIMESTAMP OF EACH SLOT OF THE SLIDING WINDOW:

  IF *(SLOT'S STATUS == SENT) AND ((CURRENT TIMESTAMP – SENT TIMESTAMP) >= TIMEOUT INTERVAL)*
  THEN PACKET HAS TO BE RESENT TO THE CLIENT, AND SLOT'S SEDNING_TIMESTAMP IS UPDATED WITH THE CURRENT_TIMESTAMP.