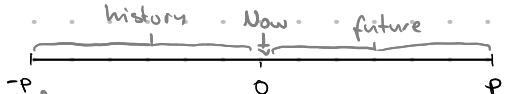
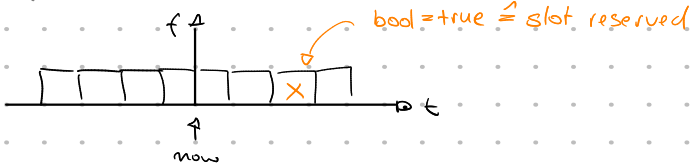
MC-SOTDMA Implementation Diary

Frequency Channel → center freq & bandwidth & is_p2p & is_blacklisted
should move to a channel blocklist!

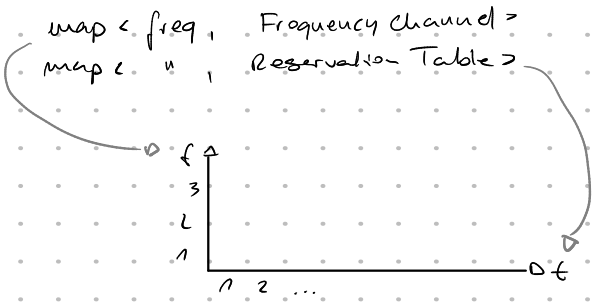
ReservationTable → std::vector<bool> utilization_vec



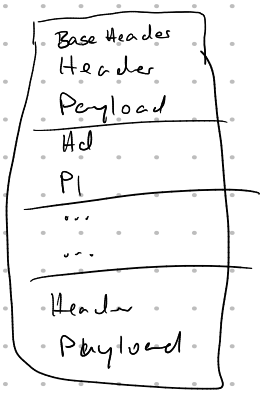


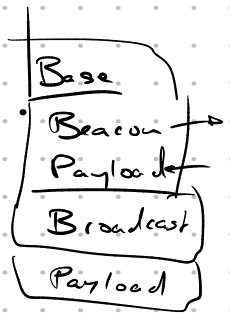
⇒ supports
isIdle(slot) isIdle(range_start, length)
findEarliestIdleRange(start, length)
mark(slot, utilized)
update(num_slots) → shifts vector to left

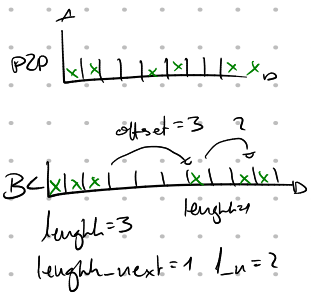
ReservationManager →



Frame Format →







Base Header

ICAO ID	27	
offset	8	
length_curr	4	+ length_next 4
timeout	8	
type	3	
CRC	16	
Σ	70	

Beacon

type
Position 12 + 14 + 12
CPR odd/even 1
#hops to GS 5
pos_quality 2

} 65

P2P usage as payload?

Broadcast

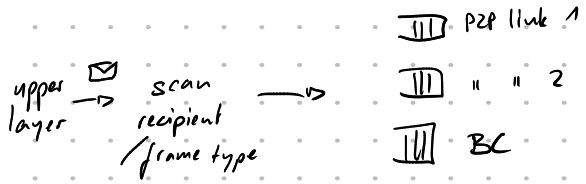
type
no additional

Unicast

type 3
destination ICAO ID 27
ARQ:
arq 1/0 1
seqno ? 8
ack_no 8
ack_slot 8

proposal as payload!

Workflow



new BC packets → BC slot selection → slot arrives → concatenate as many packets from the queue as possible

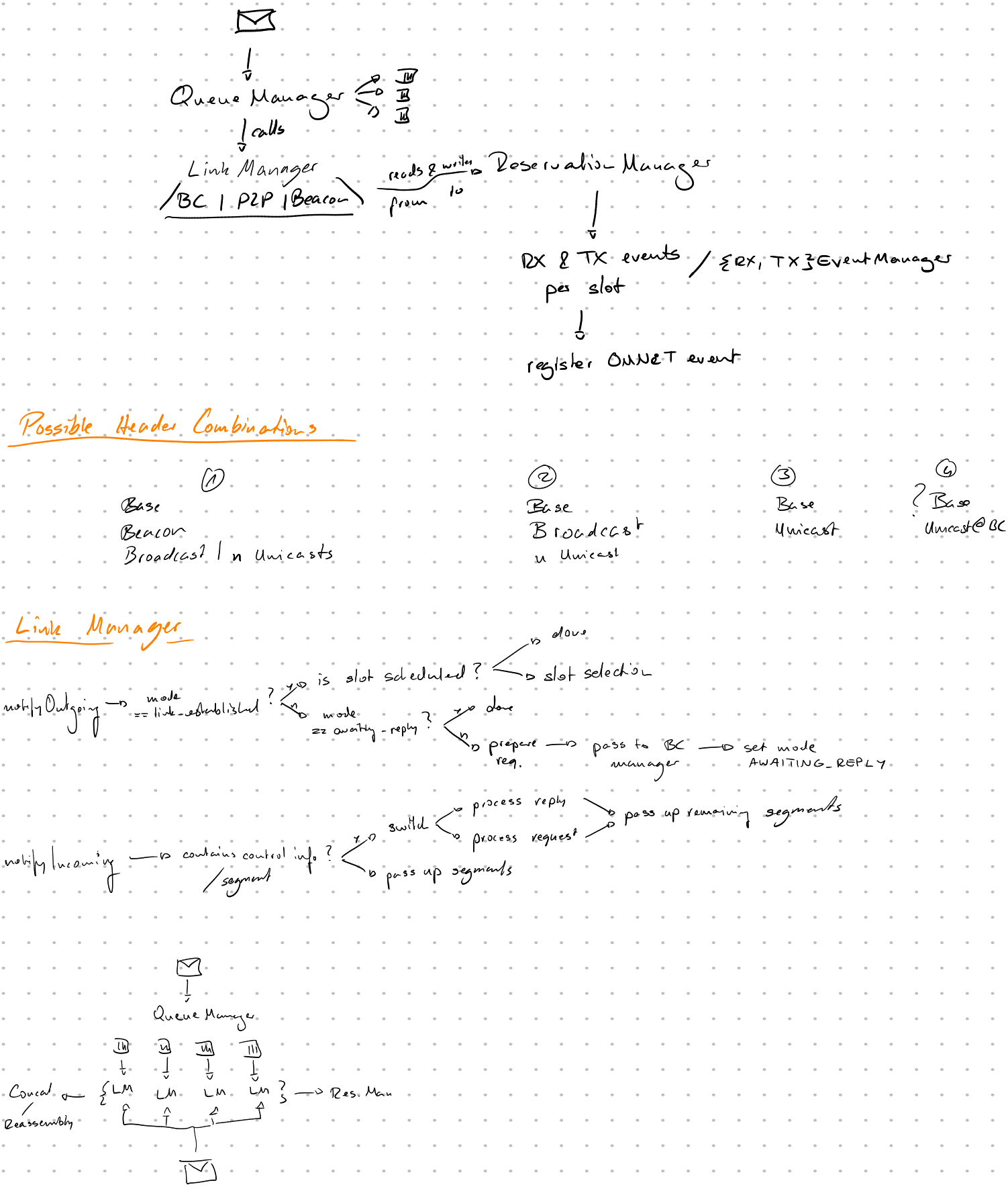
→ if more data → advertise next slot
→ don't

beacon slot @ init → arrives → put P2P map + as much broadcast data as possible → beacon slot selection

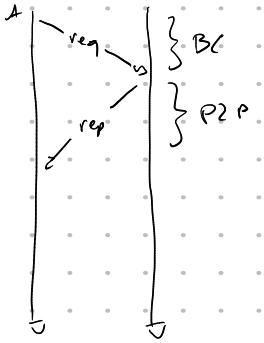
new P2P link req. → request → BC → sent → keep track of handshake

→ mark all proposal slots as RX → rcvd ✓
→ lost → rtx (w/ counter)

P2P slot arrives → concatenate as much as possible → send


Queue Manager
calls
Link Manager
BC | P2P | Beacon
reads & writes from to
Reservation Manager
RX & TX events per slot
{RX, TX} Event Manager
register OMNeT event
Possible Header Combinations
1
Base
Beacon
Broadcast | n Unicasts
2
Base
Broadcast
n Unicast
3
Base
Unicast
4
? Base
Unicast @ BC
Link Manager
notify Outgoing
mode == link-established ?
is slot scheduled ?
done
slot selection
mode == awaiting-reply ?
done
prepare req.
pass to BC manager
set mode AWAITING_REPLY
notify Incoming
contains control info / segment ?
switch
process reply
process request
pass up remaining segments
pass up segments
Queue Manager
LM
LM
LM
LM
Concat / Reassembly
Res. Man

Link Establishment Request Creation

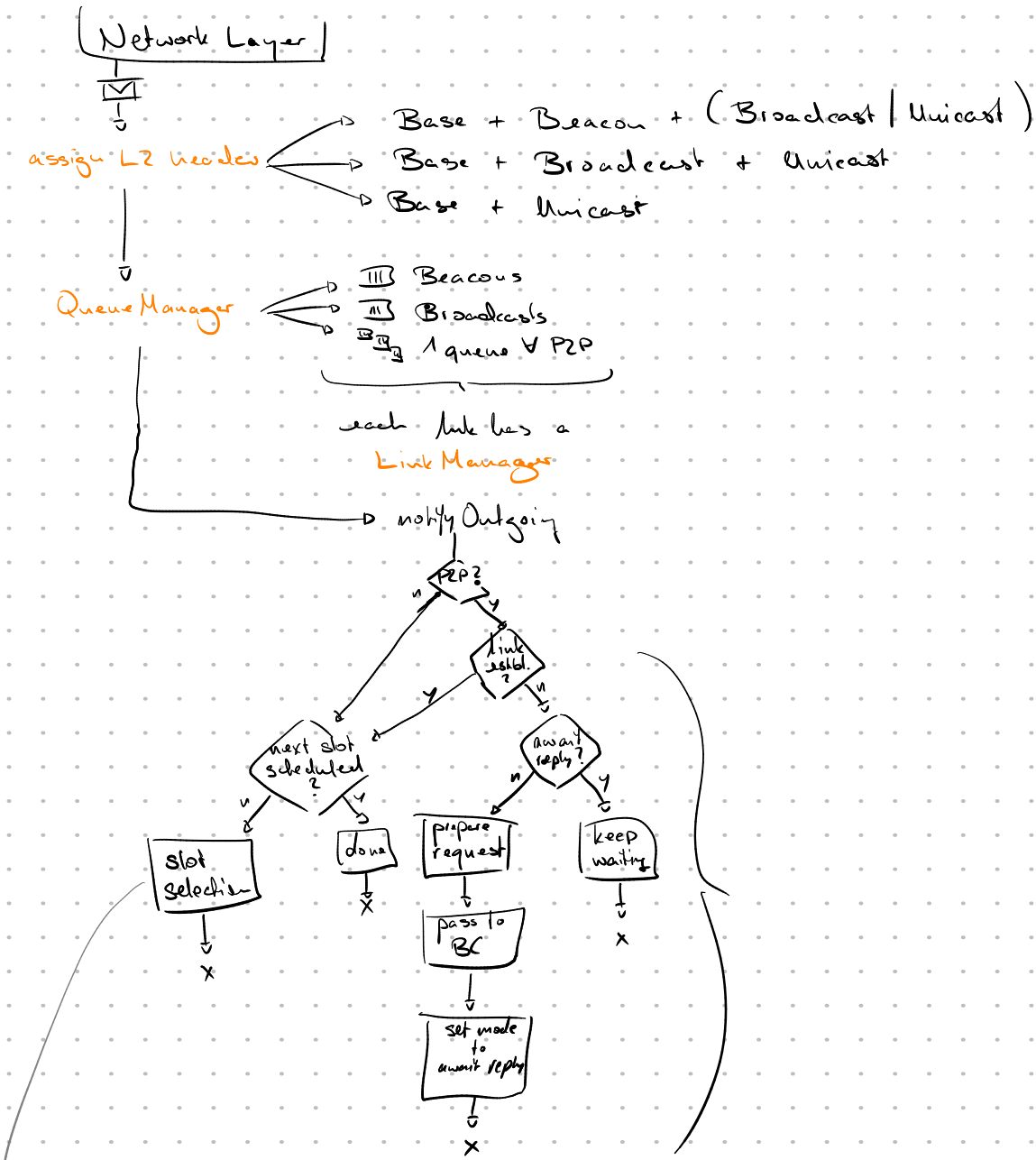


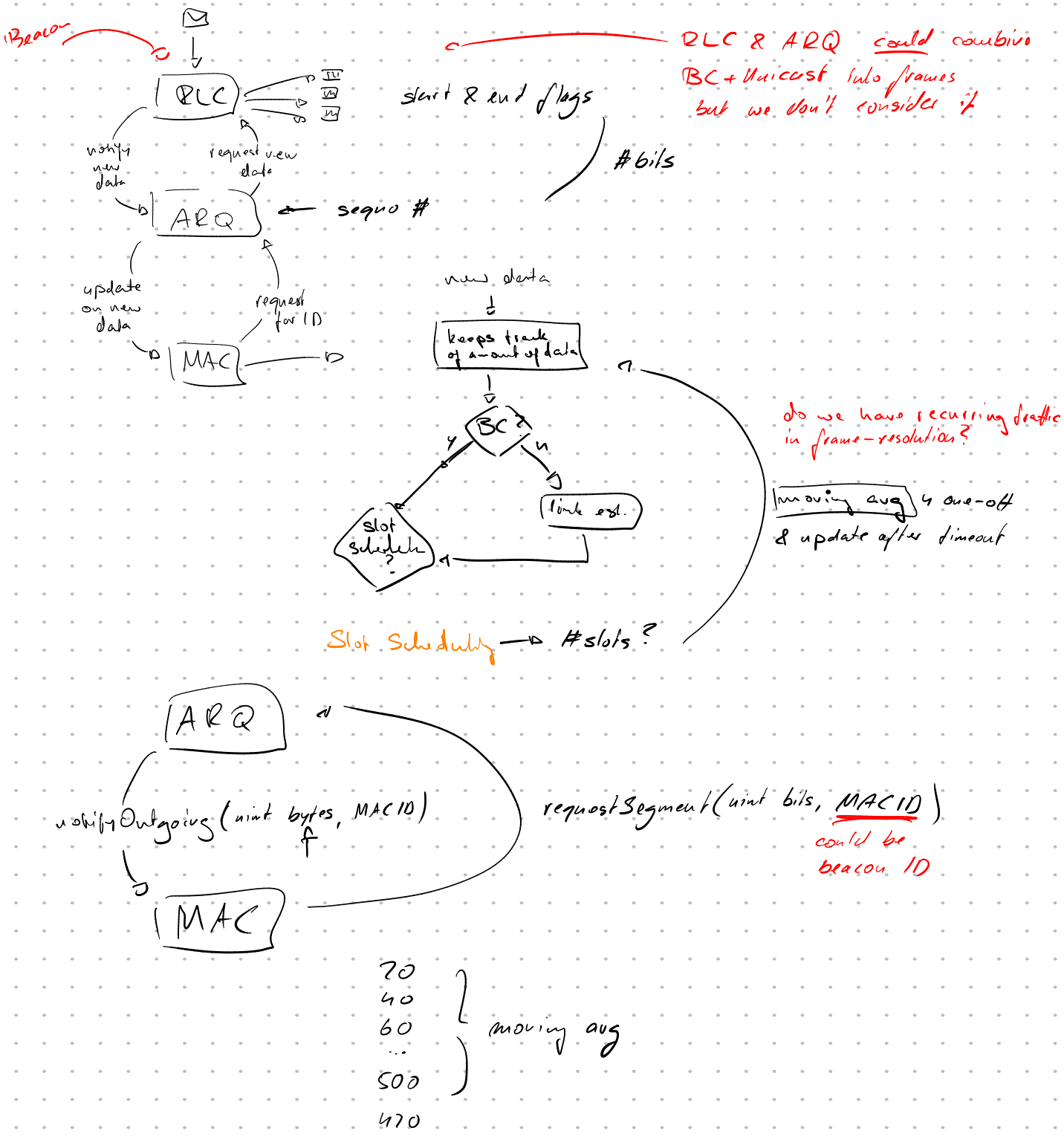
req has header unicast after a broadcast header and the proposal as payload

proposal $\{(freq, slot, length)\}$

#proposals → configurable

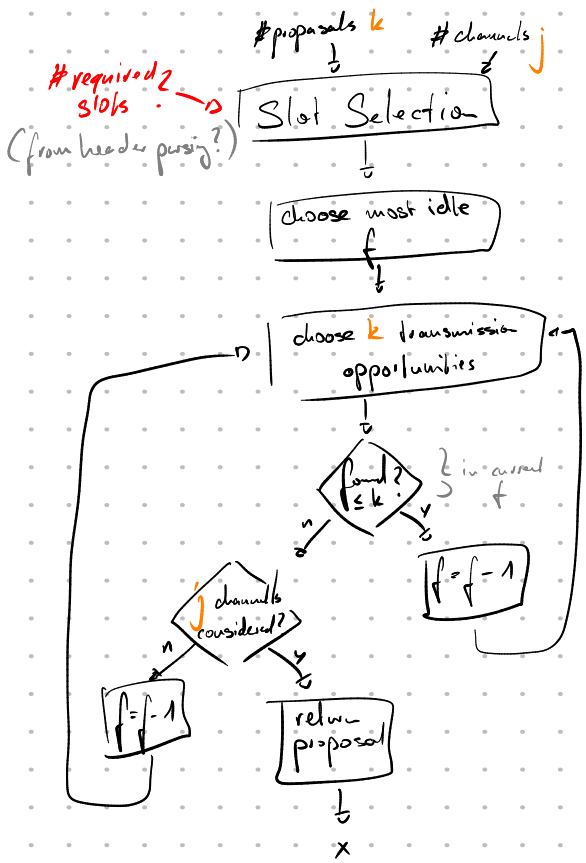
Current State of Affairs 13.11.2020



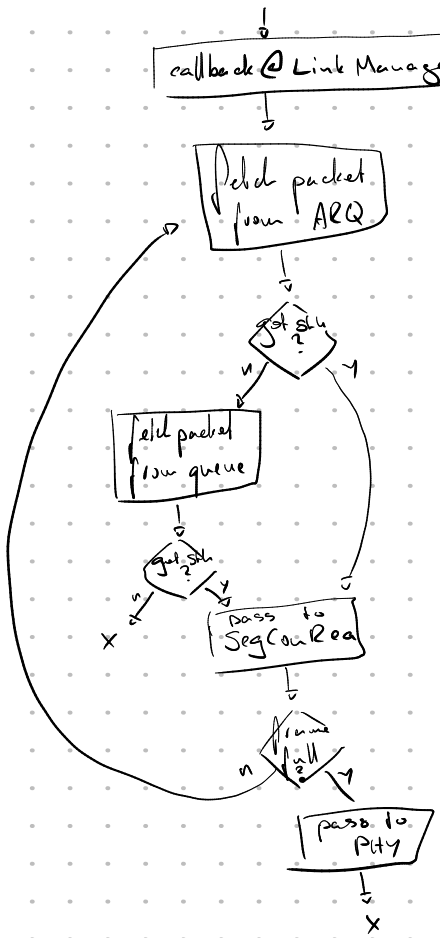

Beacon
RLC
ARQ
MAC
notify new data
request new data
sequo #
update on new data
request for ID
start & end flags
#bits
RLC & ARQ could combine
BC + Unicast into frames
but we don't consider it
new data
keeps track of amount of data
BC?
y
n
link est.
slot scheduler
do we have recurring traffic in frame-resolution?
moving avg
y one-off
& update after timeout
Slot Scheduling
#slots?
ARQ
notifyOutgoing(uint bytes, MAC ID)
requestSegment(uint bits, MAC ID)
could be beacon ID
MAC
70
40
60
...
500
470
moving avg

Reservation Manager

- BC Reservation Table (now-τ, now, now+τ)
- 1 Reservation Table ∀ P2P



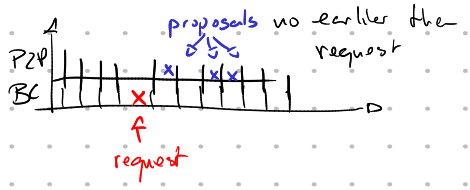
scheduled TX reservation arrives



LinkManager P2P

LinkManager BC

→ new request
→ proposal
→ slots → earliest slot?

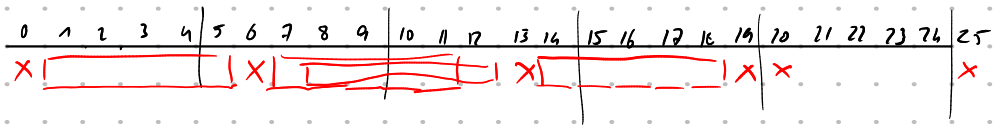


→ getNextReservation

← slot | none

request reservation →

slot ←



need new request → get BC offset
→ set proposal slots
→ set RequestPacket callback

callback → request not sent
||
request sent → update status

ARQ

void notifyOutgoing(#bits in queue, MAC ID)

L2Packet* requestSegment(#bits, MAC ID)
bool shouldLinkBeArqProtected(MAC ID)
injectPacket(L2Packet,

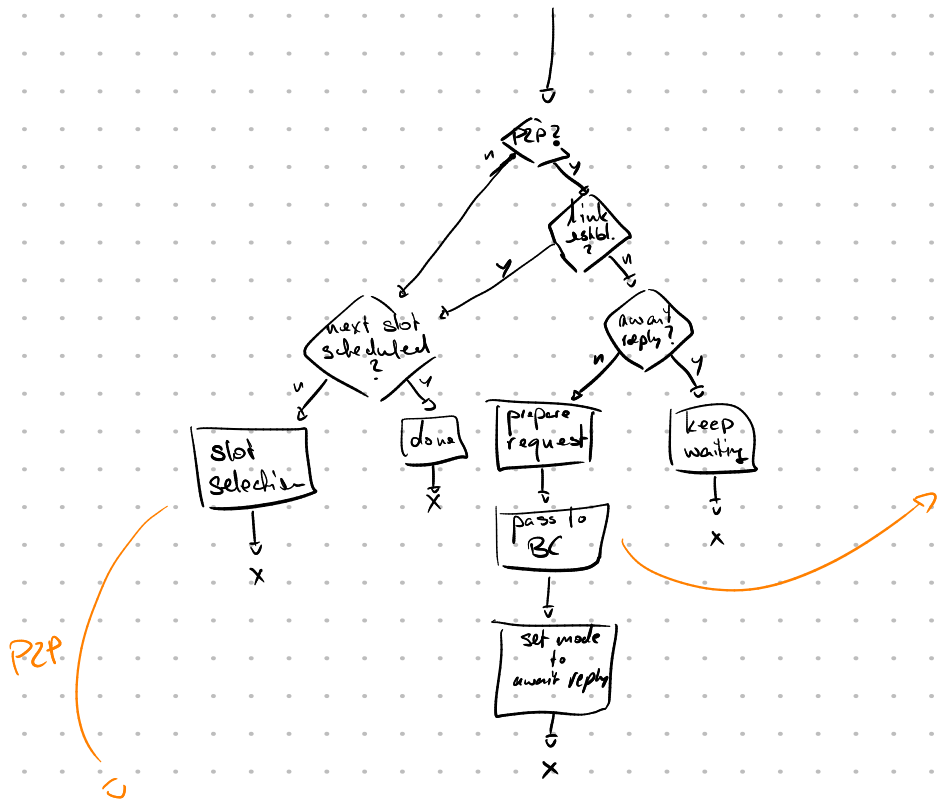
MAC

#bits, ID

ID → LinkManager

#bits

bits history: [20, 40, 60, 30, 50] → moving avg



request injected into RLC

↓

request is in BC queue

↓

slot arrives

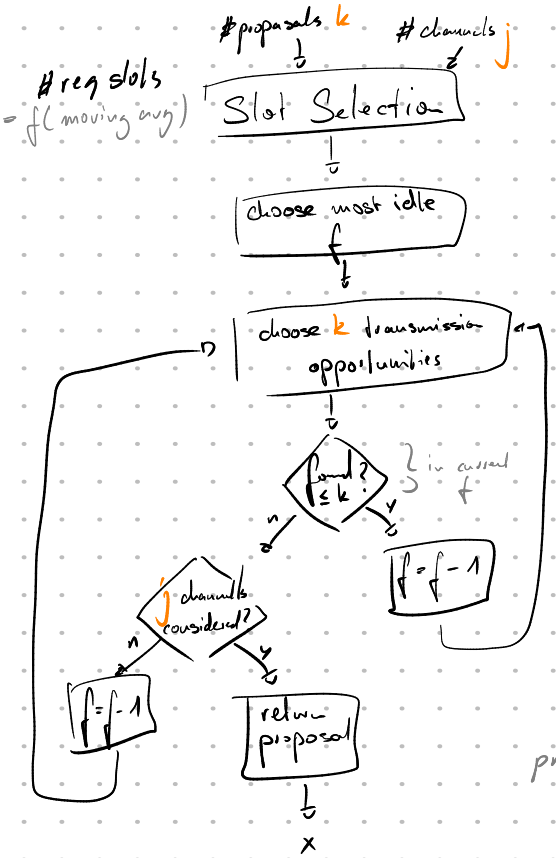
↓

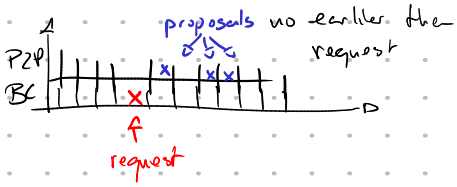
callback to original P2P Link Manager

↓

compute proposal and put into request (which must've reserved space beforehand)

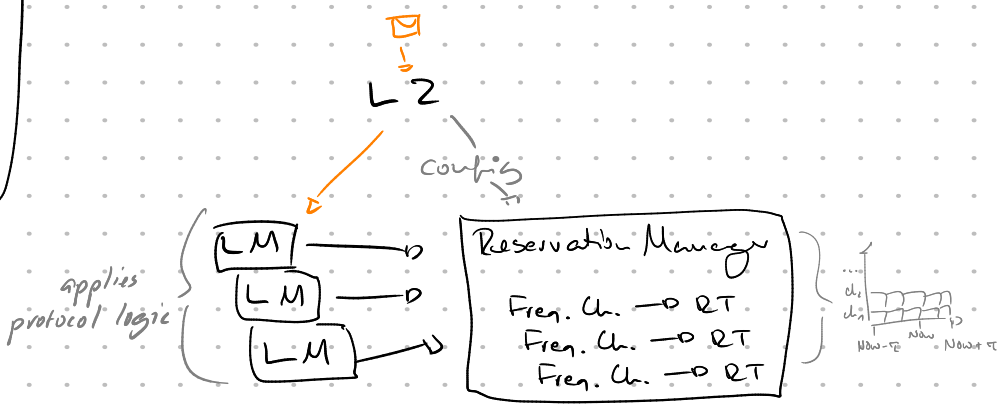
P2P



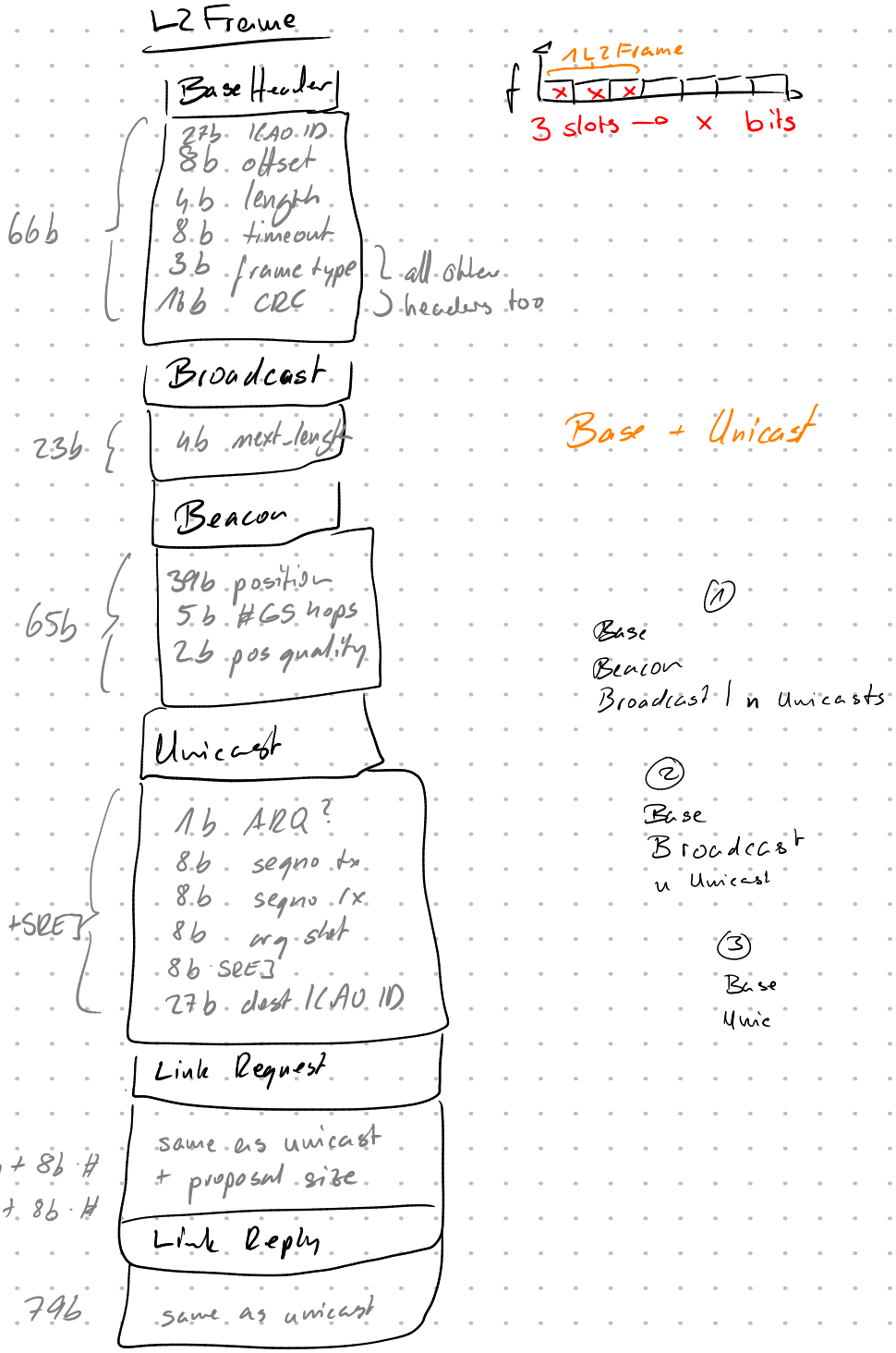


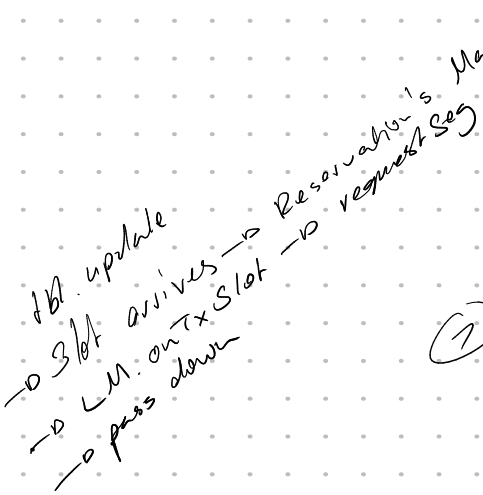
TODO BC SS

Beacon SS



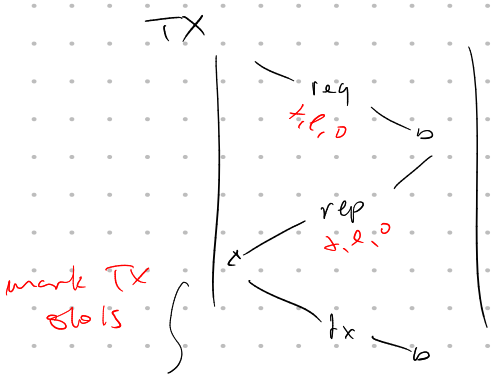
Slot arrives → LM is notified
→ PHY is queried for #bits that can be sent
→ ARQ is queried for data
→ passed down to PHY


L2 Frame
Base Header
27b ICAO ID
8b offset
4b length
8b timeout
3b frame type
16b CRC
66b
all other headers too
Broadcast
4b next_length
23b
Beacon
39b position
5b #GS hops
2b pos quality
65b
Unicast
1b ARQ?
8b seqno tx
8b seqno rx
8b arq slot
8b SEE]
27b dest ICAO ID
796 + SEE]
Link Request
same as unicast
+ proposal size
796 + 8b # + 8b #
Link Reply
same as unicast
796
L2 Frame
3 slots — o x bits
Base + Unicast
(1)
Base
Beacon
Broadcast? | n Unicasts
(2)
Base
Broadcast
u Unicast
(3)
Base
Unic



---

notifyOutgoing → requestNewLink → prepareRequest → setCallback
→ injectRLC
→ notifyOutgoing → scheduleBCSlot → mark
→ onTx → send
→ receiveFromLower → LM.rfl → parseHeaders
→ processBase
→ processRequest → prepareReply → forwardReply
→ scheduleReply TX@
mark RX @ offsets too

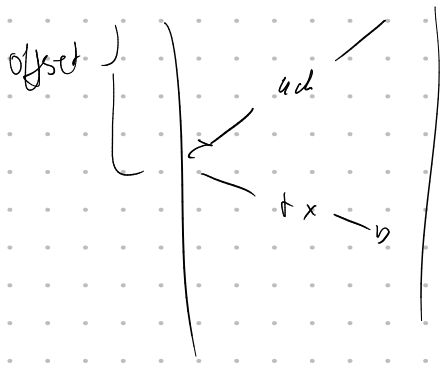


timeout, length, offset

timeout for TX!

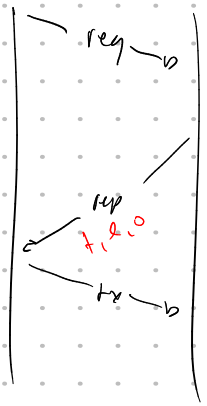
length: other users can't decode w/o knowing it

offset: dyn 4 BC fixed 4 P2P



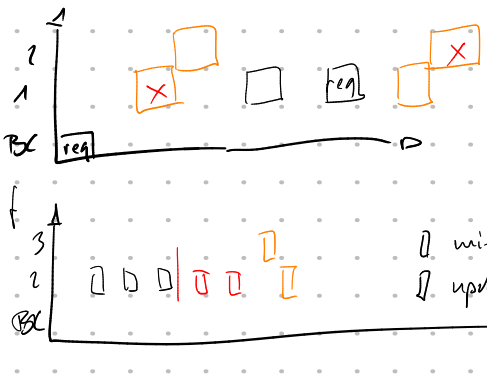
⇒ decrement timeout

data → RX ⇒ mark RX for t, l, o



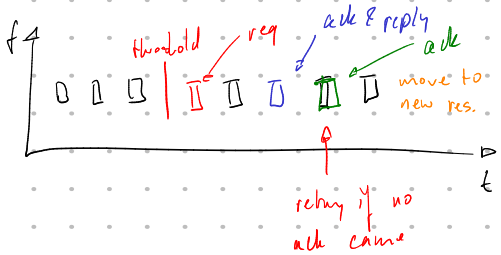
proposal → select

Link Renewal



reply in ACK

ignore RX when established



~~mark'd: either BC negotiation or force ACK-like reply~~

min offset to after expiry

① send req & await_reply

② process incoming packets w.r.t. renewal

→ detect too-late reply → trigger next try

③ get reply

→ prepare ACK

→ save chosen reserv.

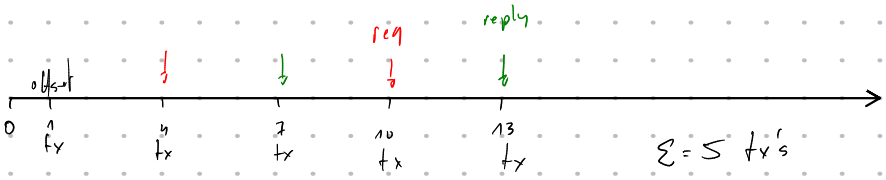
④ send ACK

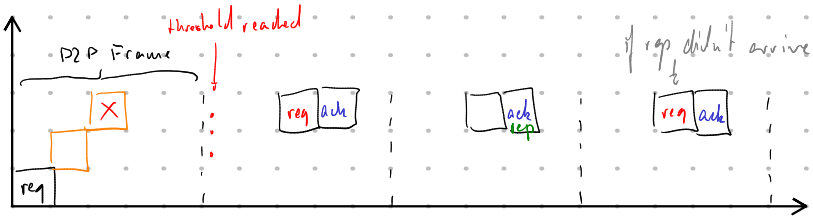
⑤ expiry → move to new reservation

ⓐ LM must know if bidirectional

ⓑ ACK expects in unicast header if uni.

ⓒ schedule control frames from header processing





✓ TX schedule: leave 1 frame for reply

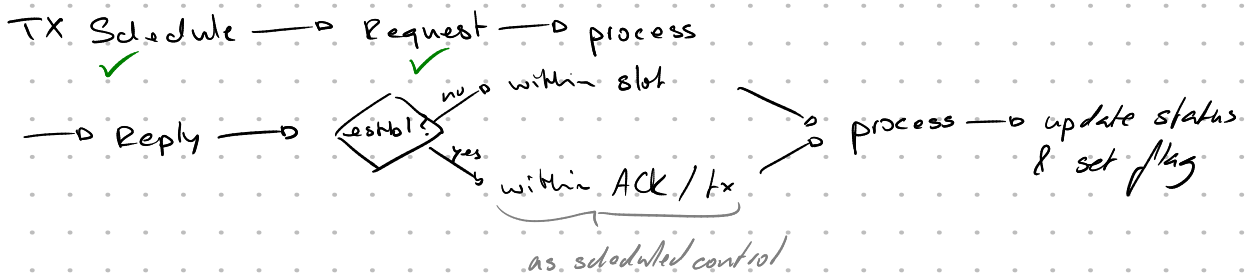
✓ process should be resettable

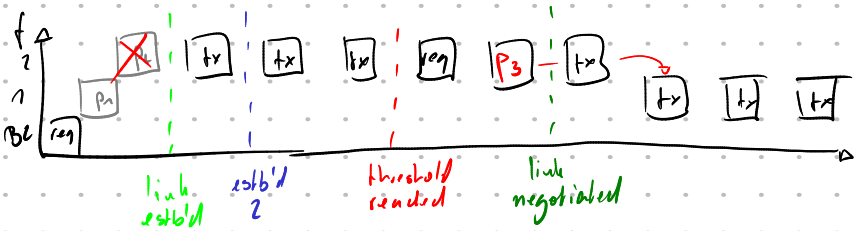
header parsing should forward replies to the process

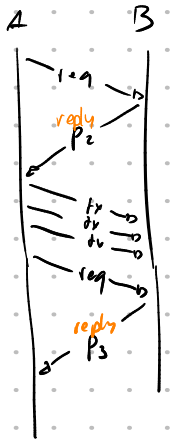
✓ process must calculate slot after which a new req is required

RX needs an ACK too (bit that indicates a new link is agreed)

actual change must be put into beacon







Tests

① estbl link, both have reservations & status

② expiring → both have status & tx has control msg

③ expiring + send req → status is reply_sent, RX has control msg

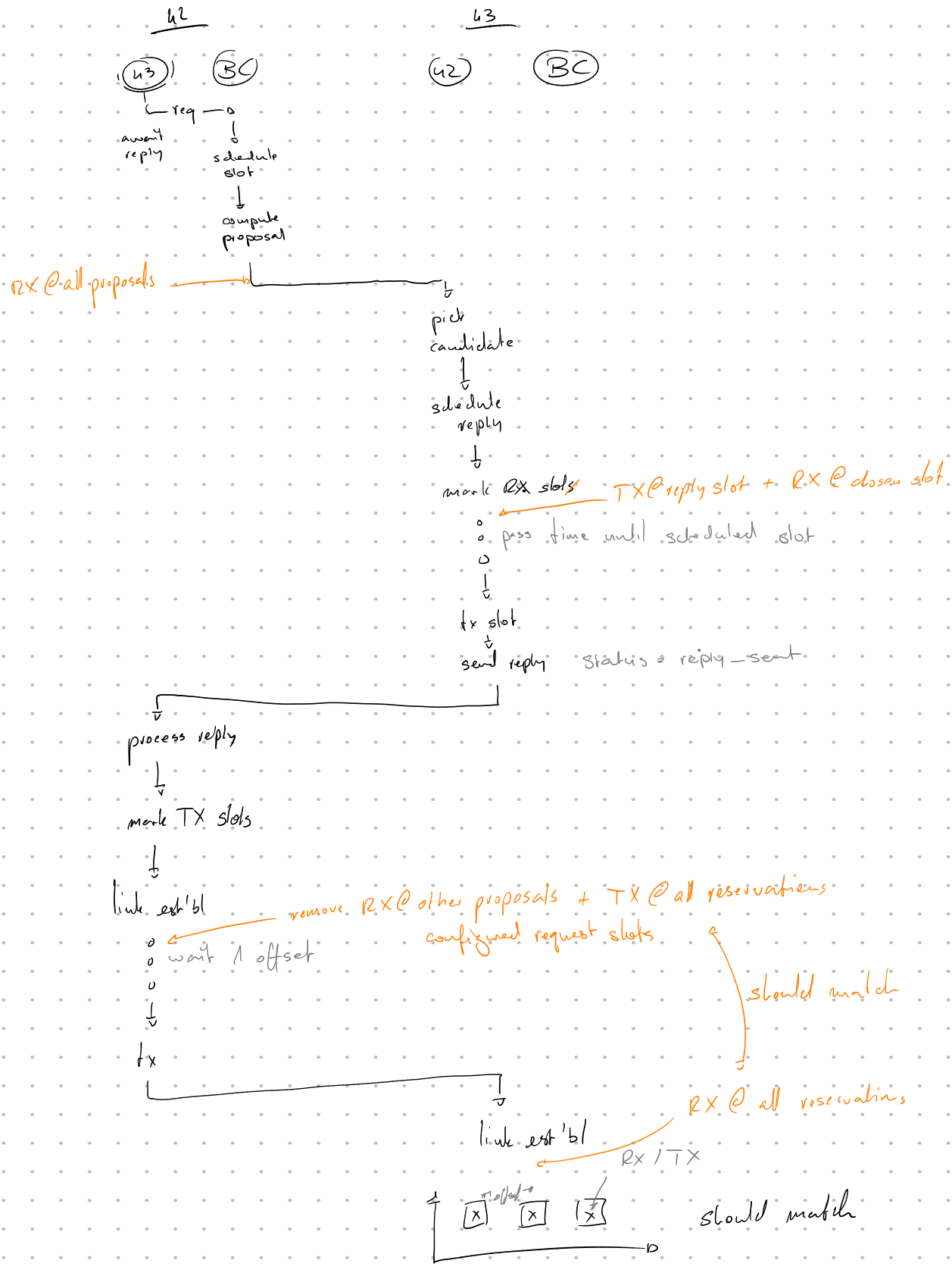
④ expiring + send rep → status is negotiated @ RX & TX

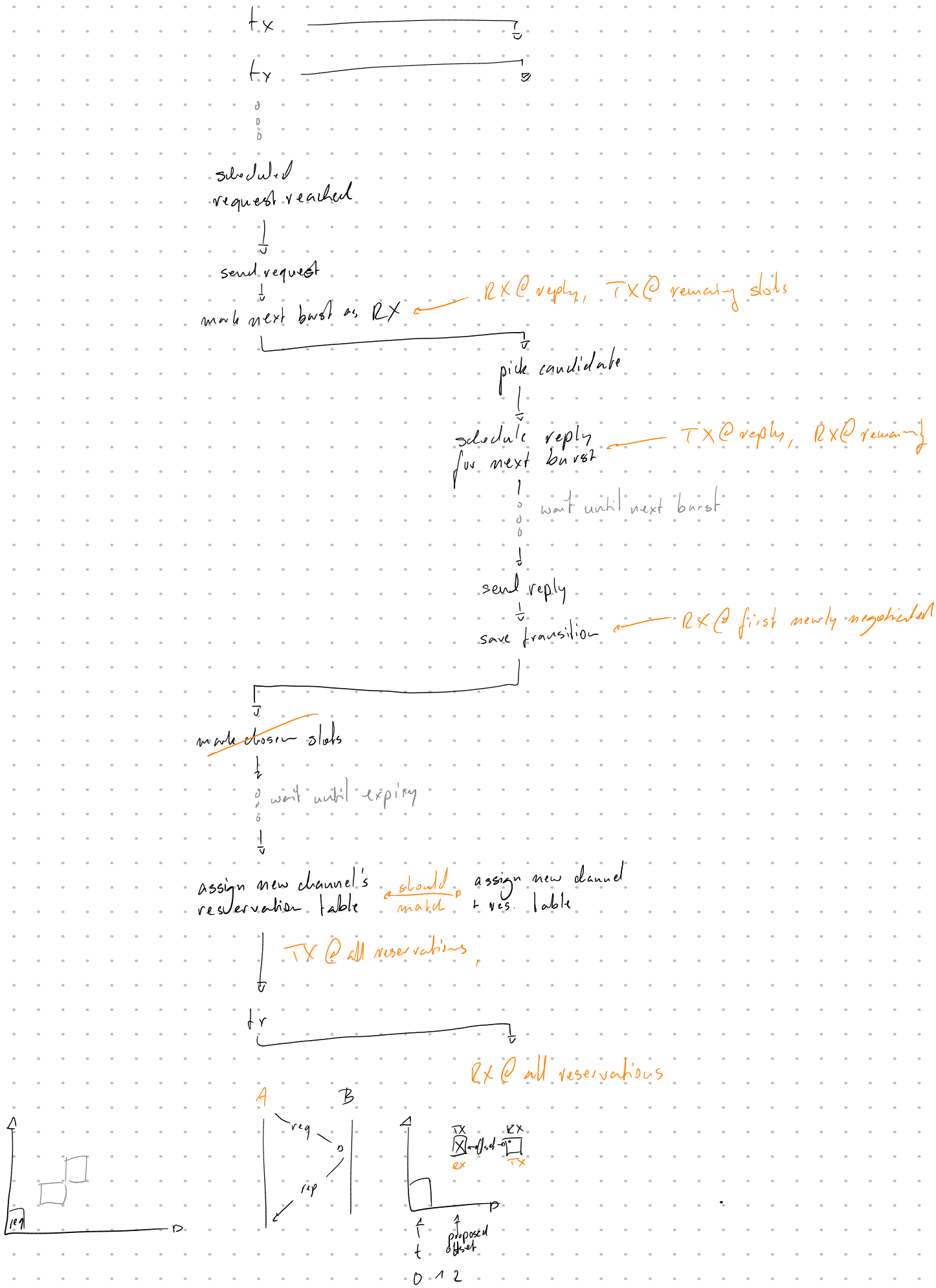
⑤ expiring + lost req → send 2nd try after not getting a reply (repeat for all links)

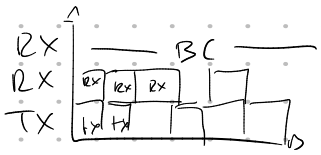
⑥ expiring + lost resp → send new req until #tries

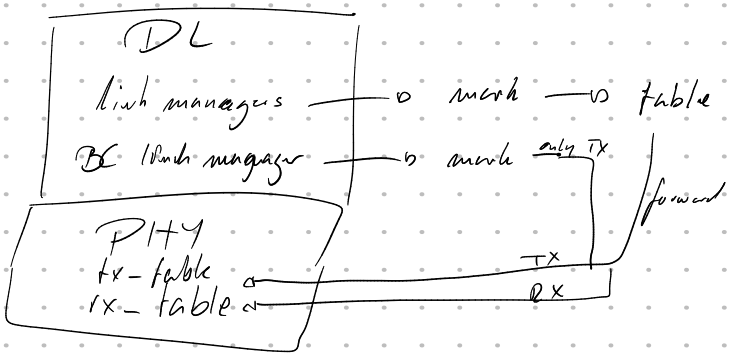
6.1 success
6.2 final fail → status is unestablished

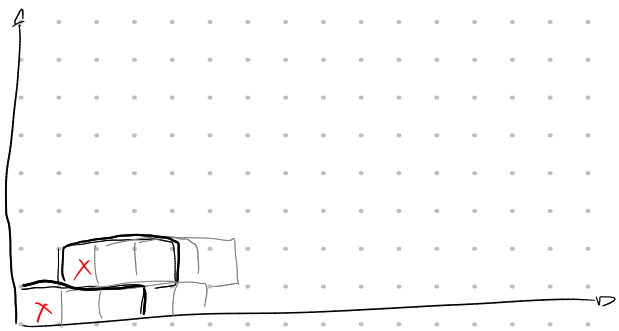
⑦ expired + new link




tx
tx
scheduled request reached
send request
mark next burst as RX
RX @ reply, TX @ remaining slots
pick candidate
schedule reply for next burst
TX @ reply, RX @ remaining
wait until next burst
send reply
save transition
RX @ first newly negotiated
mark chosen slots
wait until expiry
assign new channel's reservation table
should match
assign new channel + res. table
TX @ all reservations
tr
RX @ all reservations
A
B
req
rep
TX
RX
offset
ex
TX
proposed offset
t
0 1 2


RX
RX
TX
BC
RX
RX
RX
TX
TX


DL
link managers
match
table
BC link manager
match
only TX
forward
PHY
tx_table
rx_table
TX
RX


X
X