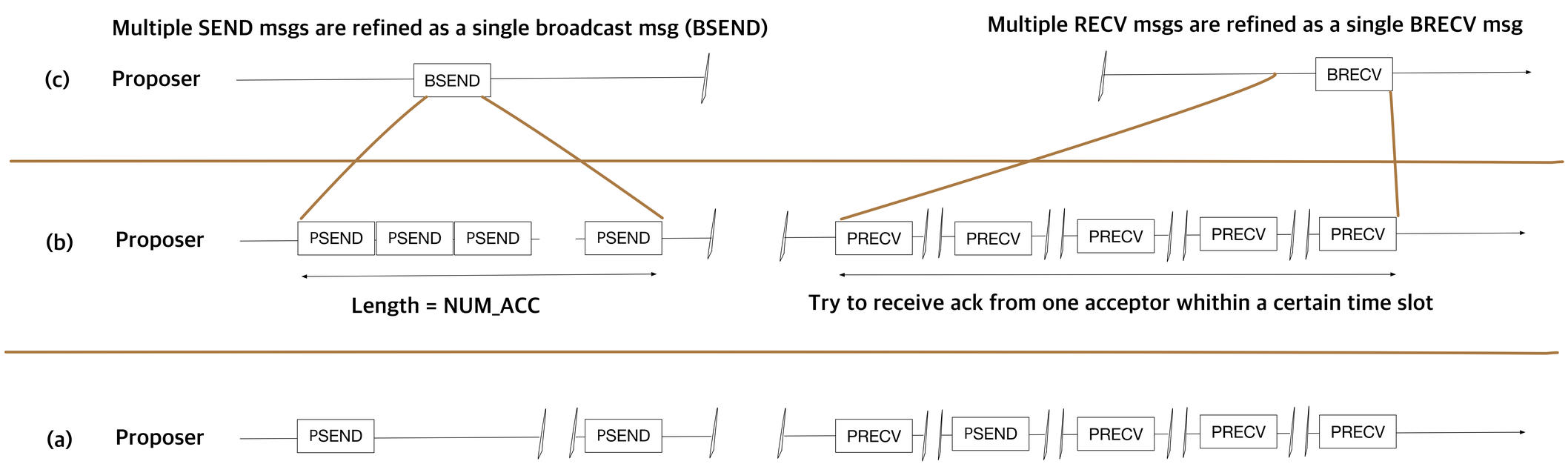


SEND Proposer -> Acceptor

PSEND (des) (msg)

BSEND (msg)

RECV Proposer <- Acceptor

PRECV (owner) (msg)

PTIME_OUT owner

BRECV (owner) (index) (addr) (prop) (msg)

BTIME_OUT (owner) (index) (addr) (prop)

Acceptor -> Proposer

ASEND (cid) (des) (msg)

Acceptor -> Proposer

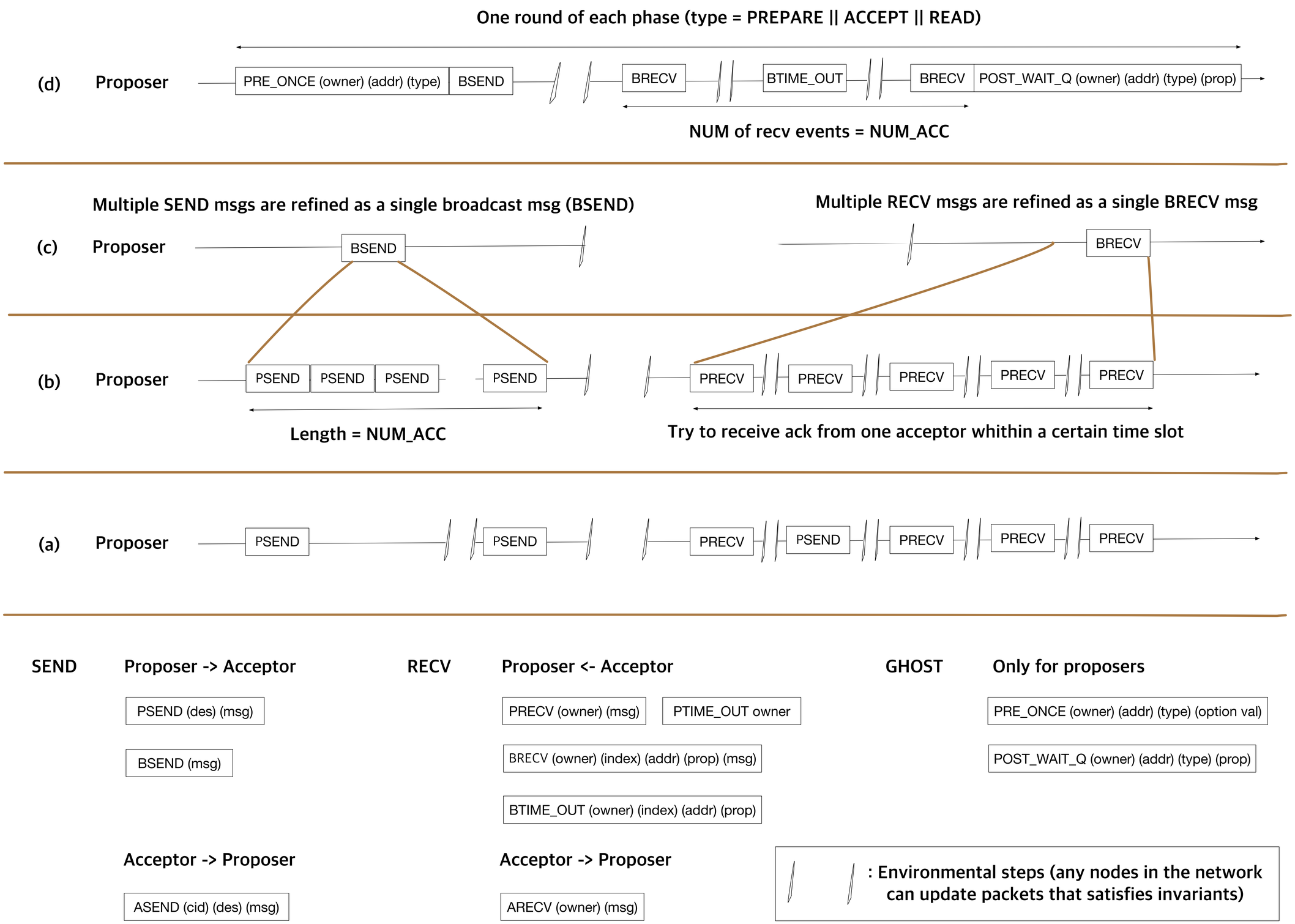
ARECV (owner) (msg)

: Environmental steps (any nodes in the network can update packets that satisfies invariants)

(c) Intermediate level - Simplify communication

(b) Implementation level - Restrict SEND and RECV pattern

(a) Interface level - Arbitrary interleaving is possible in SEND and RECV

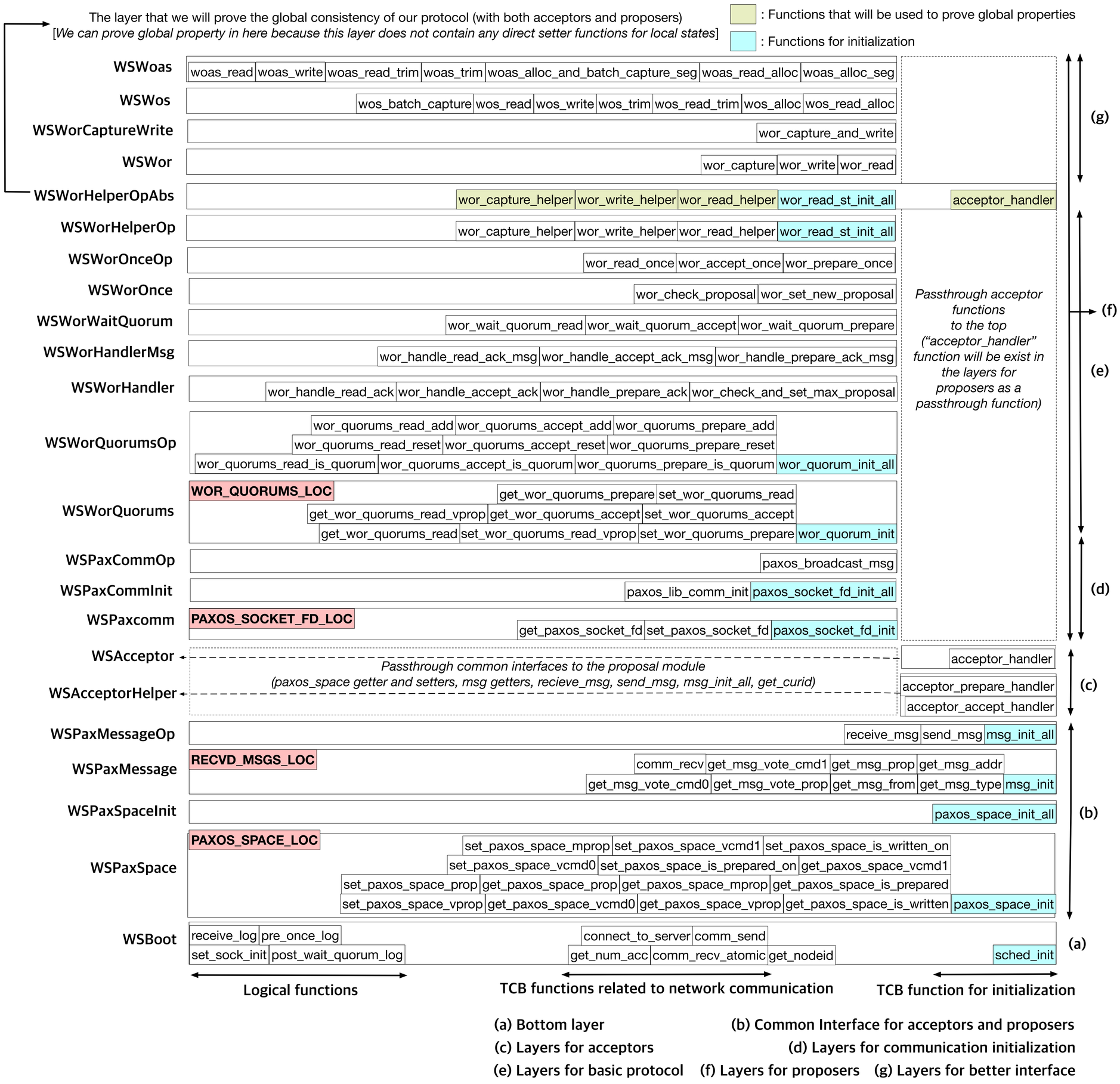


(d) Top level - Add logical events to enrich the information. All local states can be inferred by replaying network events

(c) Intermediate level - Simplify communication

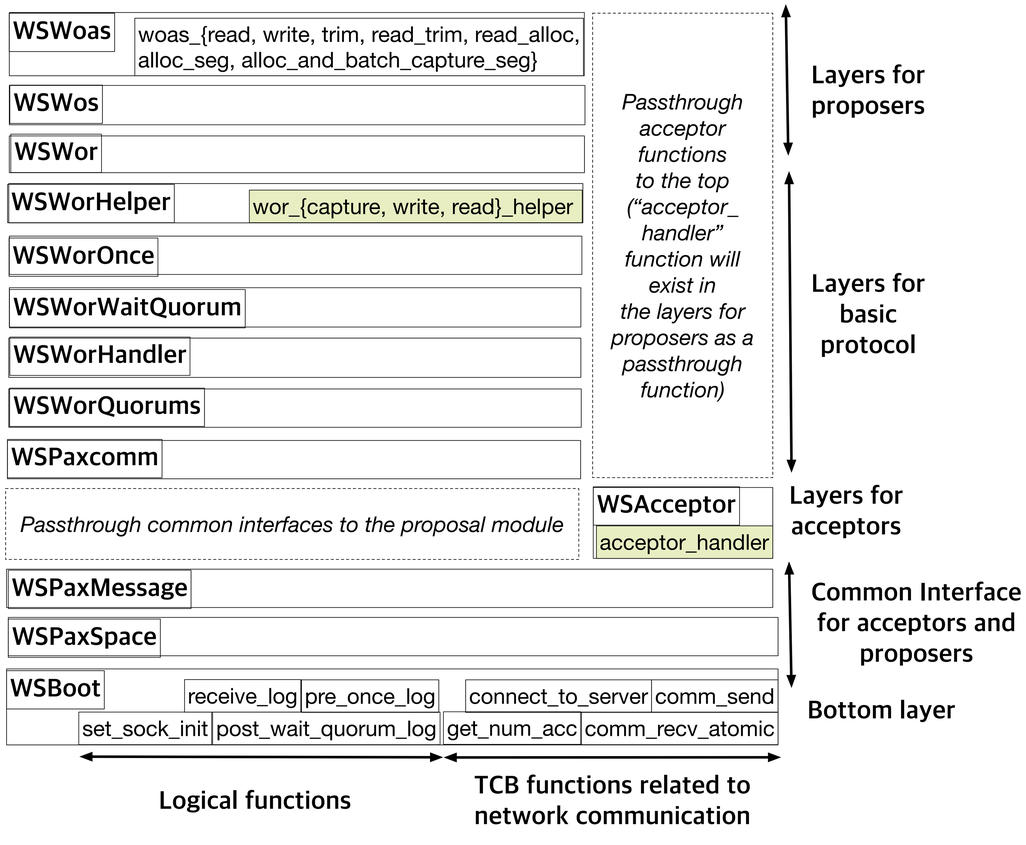
(b) Implementation level - Restrict SEND and RECV pattern

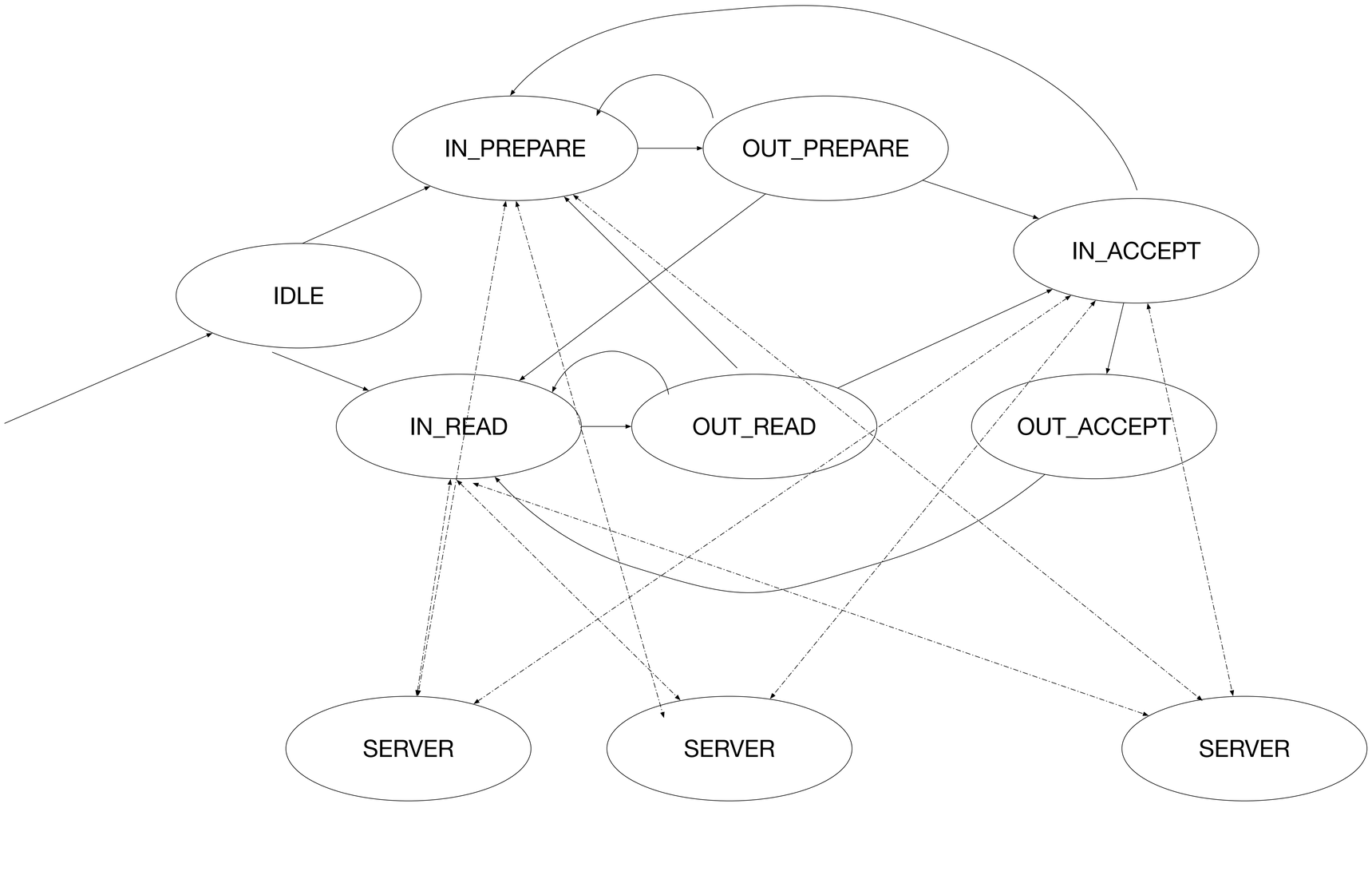
(a) Interface level - Arbitrary interleaving is possible in SEND and RECV


The layer that we will prove the global consistency of our protocol (with both acceptors and proposers)
[We can prove global property in here because this layer does not contain any direct setter functions for local states]
: Functions that will be used to prove global properties
: Functions for initialization
WSWoas
woas_read
woas_write
woas_read_trim
woas_trim
woas_alloc_and_batch_capture_seg
woas_read_alloc
woas_alloc_seg
WSWos
wos_batch_capture
wos_read
wos_write
wos_trim
wos_read_trim
wos_alloc
wos_read_alloc
WSWorCaptureWrite
wor_capture_and_write
WSWor
wor_capture
wor_write
wor_read
WSWorHelperOpAbs
wor_capture_helper
wor_write_helper
wor_read_helper
wor_read_st_init_all
acceptor_handler
WSWorHelperOp
wor_capture_helper
wor_write_helper
wor_read_helper
wor_read_st_init_all
WSWorOnceOp
wor_read_once
wor_accept_once
wor_prepare_once
WSWorOnce
wor_check_proposal
wor_set_new_proposal
WSWorWaitQuorum
wor_wait_quorum_read
wor_wait_quorum_accept
wor_wait_quorum_prepare
WSWorHandlerMsg
wor_handle_read_ack_msg
wor_handle_accept_ack_msg
wor_handle_prepare_ack_msg
WSWorHandler
wor_handle_read_ack
wor_handle_accept_ack
wor_handle_prepare_ack
wor_check_and_set_max_proposal
WSWorQuorumsOp
wor_quorums_read_add
wor_quorums_accept_add
wor_quorums_prepare_add
wor_quorums_read_reset
wor_quorums_accept_reset
wor_quorums_prepare_reset
wor_quorums_read_is_quorum
wor_quorums_accept_is_quorum
wor_quorums_prepare_is_quorum
wor_quorum_init_all
Passthrough acceptor functions to the top ("acceptor_handler" function will be exist in the layers for proposers as a passthrough function)
WSWorQuorums
WOR_QUORUMS_LOC
get_wor_quorums_prepare
set_wor_quorums_read
get_wor_quorums_read_vprop
get_wor_quorums_accept
set_wor_quorums_accept
get_wor_quorums_read
set_wor_quorums_read_vprop
set_wor_quorums_prepare
wor_quorum_init
WSPaxCommOp
paxos_broadcast_msg
WSPaxCommInit
paxos_lib_comm_init
paxos_socket_fd_init_all
WSPaxcomm
PAXOS_SOCKET_FD_LOC
get_paxos_socket_fd
set_paxos_socket_fd
paxos_socket_fd_init
WSAcceptor
acceptor_handler
Passthrough common interfaces to the proposal module
(paxos_space getter and setters, msg getters, recieve_msg, send_msg, msg_init_all, get_curid)
WSAcceptorHelper
acceptor_prepare_handler
acceptor_accept_handler
WSPaxMessageOp
receive_msg
send_msg
msg_init_all
WSPaxMessage
RECVD_MSGS_LOC
comm_recv
get_msg_vote_cmd1
get_msg_prop
get_msg_addr
get_msg_vote_cmd0
get_msg_vote_prop
get_msg_from
get_msg_type
msg_init
WSPaxSpaceInit
paxos_space_init_all
WSPaxSpace
PAXOS_SPACE_LOC
set_paxos_space_mprop
set_paxos_space_vcmd1
set_paxos_space_is_written_on
set_paxos_space_vcmd0
set_paxos_space_is_prepared_on
get_paxos_space_vcmd1
set_paxos_space_prop
get_paxos_space_prop
get_paxos_space_mprop
get_paxos_space_is_prepared
set_paxos_space_vprop
get_paxos_space_vcmd0
get_paxos_space_vprop
get_paxos_space_is_written
paxos_space_init
WSBoot
receive_log
pre_once_log
set_sock_init
post_wait_quorum_log
connect_to_server
comm_send
get_num_acc
comm_recv_atomic
get_nodeid
sched_init
(g)
(f)
(e)
(d)
(c)
(b)
(a)
Logical functions
TCB functions related to network communication
TCB function for initialization
(a) Bottom layer
(b) Common Interface for acceptors and proposers
(c) Layers for acceptors
(d) Layers for communication initialization
(e) Layers for basic protocol
(f) Layers for proposers
(g) Layers for better interface

The layer that we will prove the global consistency of our protocol (with both acceptors and proposers)
[*We can prove global property in here because this layer does not contain any direct setter functions for local states*]

: Functions that will be used to prove global properties




IN_PREPARE
OUT_PREPARE
IN_ACCEPT
IDLE
IN_READ
OUT_READ
OUT_ACCEPT
SERVER
SERVER
SERVER