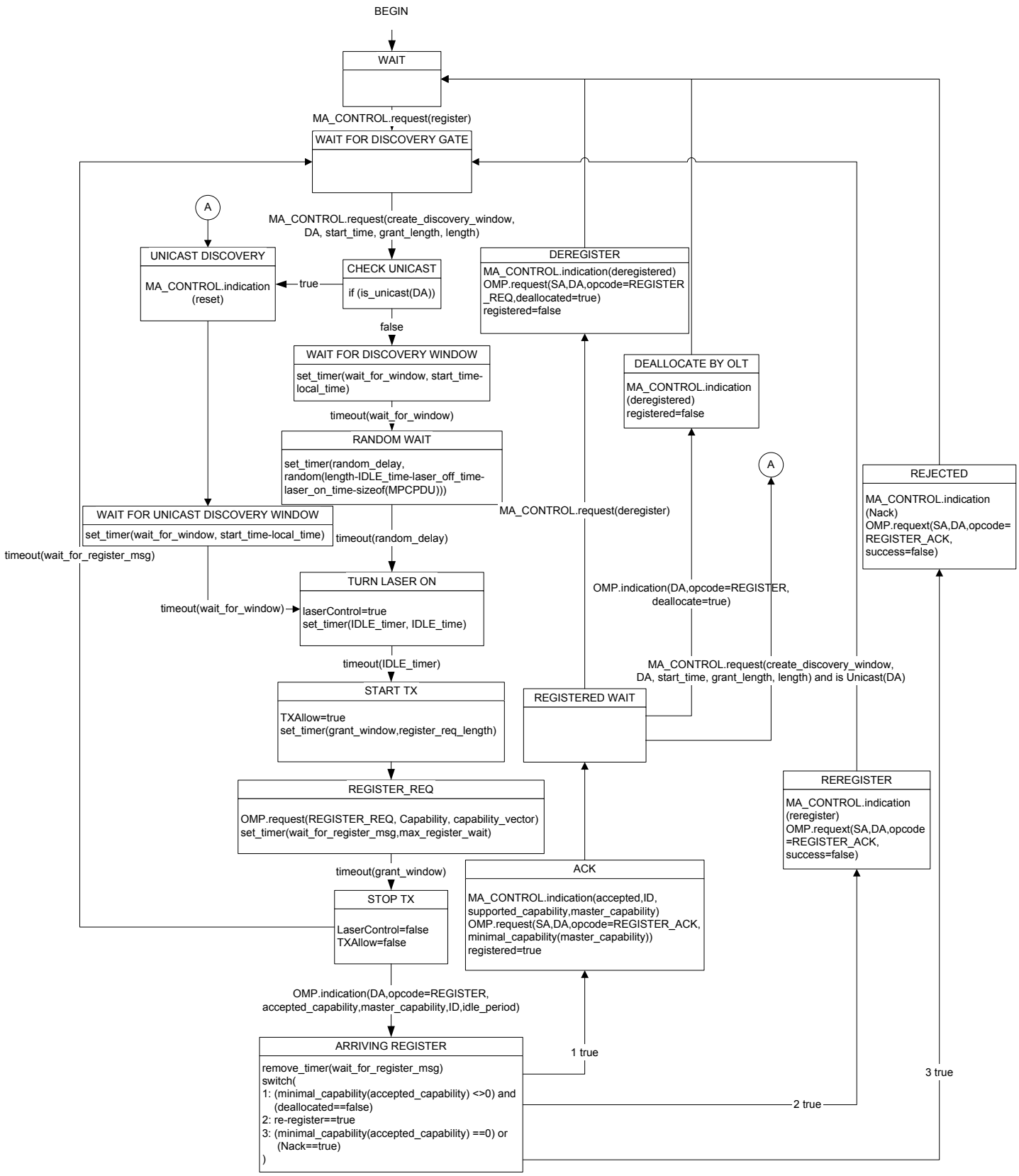
BEGIN
WAIT
MA_CONTROL.request(register)
WAIT FOR DISCOVERY GATE
A
MA_CONTROL.request(create_discovery_window, DA, start_time, grant_length, length)
UNICAST DISCOVERY
MA_CONTROL.indication (reset)
true
CHECK UNICAST
if (is_unicast(DA))
DEREGISTER
MA_CONTROL.indication(deregistered)
OMP.request(SA,DA,opcode=REGISTER_REQ,deallocated=true)
registered=false
false
WAIT FOR DISCOVERY WINDOW
set_timer(wait_for_window, start_time-local_time)
DEALLOCATE BY OLT
MA_CONTROL.indication (deregistered)
registered=false
timeout(wait_for_window)
RANDOM WAIT
set_timer(random_delay, random(length-IDLE_time-laser_off_time-laser_on_time-sizeof(MPCPDU)))
A
REJECTED
MA_CONTROL.indication (Nack)
OMP.requext(SA,DA,opcode=REGISTER_ACK, success=false)
MA_CONTROL.request(deregister)
WAIT FOR UNICAST DISCOVERY WINDOW
set_timer(wait_for_window, start_time-local_time)
timeout(random_delay)
timeout(wait_for_register_msg)
TURN LASER ON
OMP.indication(DA,opcode=REGISTER, deallocate=true)
timeout(wait_for_window)
laserControl=true
set_timer(IDLE_timer, IDLE_time)
timeout(IDLE_timer)
MA_CONTROL.request(create_discovery_window, DA, start_time, grant_length, length) and is Unicast(DA)
START TX
TXAllow=true
set_timer(grant_window,register_req_length)
REGISTERED WAIT
REREGISTER
MA_CONTROL.indication (reregister)
OMP.requext(SA,DA,opcode=REGISTER_ACK, success=false)
REGISTER_REQ
OMP.request(REGISTER_REQ, Capability, capability_vector)
set_timer(wait_for_register_msg,max_register_wait)
timeout(grant_window)
ACK
MA_CONTROL.indication(accepted,ID, supported_capability,master_capability)
OMP.request(SA,DA,opcode=REGISTER_ACK, minimal_capability(master_capability))
registered=true
STOP TX
LaserControl=false
TXAllow=false
OMP.indication(DA,opcode=REGISTER, accepted_capability,master_capability,ID,idle_period)
ARRIVING REGISTER
remove_timer(wait_for_register_msg)
switch(
1: (minimal_capability(accepted_capability) <>0) and (deallocated==false)
2: re-register==true
3: (minimal_capability(accepted_capability) ==0) or (Nack==true)
)
1 true
2 true
3 true