DESERT Underwater

User Reference Manual

Roberto Francescon <roberto.francescon@wirelessandmore.it>

December 17, 2021

Contents

1 Introduction

This document is not a user manual, this document aims at being a complete reference for all the available settings that can be used in a Tcl script of a DESERT simulation.These settings can be divided into two categories: bound variables and commands: please see one of the simulation examples for their usage. The structure of each heading is the following: the first name, in monospace, is the name of the Tcl module that can be deployed in a simulation, then, in parentheses and in italic, is the C++ class that is mapped by this module and then after the comma, still in italic, is the father class of this C++ class. In general, everything in monospace refers to Tcl elements and everything in italic refers to C++ elements. For what concern the bound variables, their Tcl name is presented first, in monospace, then in parentheses and in italic, their bound C++ class member is found.

2 physical layer

2.1 `Module/UW/MPhypatch` (*UWMPhypatch*) : *MPhy*

2.1.1 Bound variables

1. `debug_` (*debug_*)

Flag to enable debug mode (i.e., printing of debug messages) if set to 1.

2.1.2 Commands

2.2 Module/UW/UwModem/AHOI (*UwAhoiModem*) : *UwModem*

2.2.1 Bound variables

1. `buffer_size` (*DATA_BUFFER_LEN*)
   Size of the buffer that holds data
2. `max_read_size` (*MAX_READ_BYTES*)
   Maximum number of bytes to be read by a single dump of data
3. `parity_bit` (*parity_bit*)
   flag for parity bit
4. `stop_bit` (*stop_bit*)
   flag for stop bit
5. `flow_control` (*flow_control*)
   flag for flow control
6. `baud_rate` (*baud_rate*)
   Integer for port baud rate
7. `modem_id` (*id*)
   Id of a parallel thread.
8. `max_n_retx` (*MAX_RETX*)
   Maximum number of retransmissions for the same packet
9. `wait_delivery` (*WAIT_DELIVERY_INT*)
   Time interval matching the WAIT_DELIVERY variable: version of type int to match the chrono one, needed because TclObject::bind does not support binding std::chrono variable [milliseconds]

2.2.2 Commands

2.3 Module/UW/GAINFROMDB (*UnderwaterGainFromDb*) : *UnderwaterPhysical*

2.3.1 Bound variables

1. time_roughness_ (*time_ roughness_*)

   Roughness of the temporal samples.

2. depth_roughness_ (*depth_ roughness_*)

   Roughness of the depth samples.

3. distance_roughness_ (*distance_ roughness_*)

   Roughness of the distance samples.

4. total_time_ (*total_ time_*)

   Maximum value of the temporal samples, after this limit the smilulation time will be reset to zero.

5. frequency_correction_factor_ (*frequency_ correction_ factor_*)

   used to shift from a frequency value to another one.

2.3.2 Commands

1. path

2.4 Module/UW/MPhy_modem/S2CLowLevel (*MS2C_ Evo_ lowlev*) : *UWMPhy_ modem*

2.4.1 Bound variables

1. bitrate_index (*bitrate_ i*)

   Variable holding the bitrate index of the low level firmware

2. SL (*SL*)

   Variable holding the Source Level of the low level driver

3. msg_bitlength (*msg_ bitlen*)

   Very very temporary parameter to let the receiver not screw up and read only the, known, number of bytes

2.4.2 Commands

2.5 Module/UW/PROPAGATIONROGERS (*UnderwaterPhysicalRogersModel*) : *UnderwaterMPropagation*

2.5.1 Bound variables

1. bottom_depth_ (*bottom_ depth*)
   Water depth (m)
2. sound_speed_water_bottom_ (*sound_ speed_ water_ bottom*)
   Speed of sound in water at the sea bottom level (m/s).
3. sound_speed_water_surface_ (*sound_ speed_ water_ surface*)
   Speed of sound in water at the sea surface level (m/s).
4. sound_speed_sediment_ (*sound_ speed_ sediment*)
   Speed of sound in the sediment (m/s).
5. density_sediment_ (*density_ sediment*)
   Sediment density (g/cm^3).
6. density_water_ (*density_ water*)
   Water density (g/cm^3).
7. attenuation_coeff_sediment_ (*attenuation_ coeff_ sediment*)
   *Attenuation coefficient of the sediment (dB/(m*kHz)).*
8. debug_ (*debug_*)
   Flag to enable debug mode (i.e., printing of debug messages) if set to 1.

2.5.2 Commands

1. getBottomDepth
2. getSoundSpeedWaterBottom
3. getSoundSpeedWaterSurface
4. getSoundSpeedSediment
5. getDensitySediment

6. `getDensityWater`
7. `getAttenuationCoeffSediment`
8. `setBottomDepth`
9. `setSoundSpeedWaterBottom`
10. `setSoundSpeedWaterSurface`
11. `setSoundSpeedSediment`
12. `setDensitySediment`
13. `setDensityWater`
14. `setAttenuationCoeffSediment`

2.6 `Module/UW/AHOI/PHY` (*UwAhoiPhy*) : //

2.6.1 Bound variables

2.6.2 Commands

1. `initLUT`
2. `setRangePDRFileName`
3. `setSIRFileName`
4. `setLUTSeparator`

2.7 `Module/UW/UWOPTICALBEAMPATTERN` (*UwOpticalBeamPattern*) : *UwOpticalPhy*

2.7.1 Bound variables

1. `noise_threshold` (*back_ noise_ threshold_*)
2. `inclination_angle_` (*inclination_ angle_*)
   Angle of inclination from the 0 Zenith

2.7.2 Commands

1. `useSameBeamPattern`
2. `useDifferentBeamPattern`
3. `setBeamPatternPath`
4. `setMaxRangePath`
5. `setBeamSeparator`
6. `setMaxRangeSeparator`
7. `setInclinationAngle`
8. `setBeamPatternPath`

2.8 `Module/UW/PHYSICAL` (*UnderwaterPhysical*) : *UnderwaterMPhyBpsk*

2.8.1 Bound variables

1. `rx_power_consumption_` (*rx_power_*)
   Power required in reception.
2. `tx_power_consumption_` (*tx_power_*)
   Power required in transmission.

2.8.2 Commands

1. `getTxTime`
2. `getRxTime`
3. `getConsumedEnergyTx`
4. `getConsumedEnergyRx`
5. `getTransmittedBytes`
6. `getTotPktsLost`
7. `getCollisionsDATAvsCTRL`
8. `getCollisionsCTRL`

9. `getCollisionsDATA`
10. `getTotCtrlPktsLost`
11. `getErrorCtrlPktsInterf`
12. `modulation`
13. `setInterferenceModel`
14. `setInterference`

2.9 `Module/UW/HERMES/PHY` (*UwHermesPhy*) : *UnderwaterPhysical*

2.9.1 Bound variables

1. `BCH_N` (*BCH_T*)
2. `FRAME_BIT` (*FRAME_BIT*)

2.9.2 Commands

1. `initLUT`
2. `setLUTFileName`
3. `setLUTSeparator`

2.10 `Module/UW/UwModem/EvoLogicsS2C` (*UwEvoLogicsS2CModem*) : *UwModem*

2.10.1 Bound variables

1. `buffer_size` (*DATA_BUFFER_LEN*)

   Size of the buffer that holds data

2. `max_read_size` (*MAX_READ_BYTES*)

   Maximum number of bytes to be read by a single dump of data

3. `max_n_status_queries` (*MAX_N_STATUS_QUERIES*)

   *Maximum number of time to query the modem transmission status before to * discard the transmitted packet*

2.10.2 Commands

1. `start`
2. `stop`
3. `setBurstMode`
4. `setIMMode`
5. `enableIMAck`
6. `disableIMAck`
7. `setSourceLevel`

2.11 `Module/UW/PHYSICALFROMDB` (*UnderwaterPhysicalfromdb*) : *UnderwaterGainFromDb*

2.11.1 Bound variables

1. `tau_index_` (*tau_ index*)

   Tau index to load in the file.

2.11.2 Commands

1. `setPathGainmaps`
2. `setPathSelfInterference`

2.12 `Module/UW/MPhy_modem/S2C` (*MS2C_ EvoLogics*) : *UWM-Phy_ modem*

2.12.1 Bound variables

1. `UseKeepOnline_` (*UseKeepOnline*)
2. `NoiseProbeFrequency_` (*NoiseProbeFrequency*)
3. `MultipathProbeFrequency_` (*MultipathProbeFrequency*)
4. `DeafTime_` (*DeafTime*)

2.12.2 Commands

2.13 UW/AL/Packer (*packer*) : *TclObject*

2.13.1 Bound variables

1. debug_ (*debug_*)

   Flag to enable debug mode (i.e., printing of debug messages) if set to 1.

2. SRC_ID_Bits (*SRC_ ID_ Bits*)

3. PKT_ID_Bits (*PKT_ ID_ Bits*)

   Bit length of the srcID_ field to be put in the header stream of bits.

4. FRAME_OFFSET_Bits (*FRAME_ OFFSET_ Bits*)

   Bit length of the pktID_ field to be put in the header stream of bits.

5. M_BIT_Bits (*M_ BIT_ Bits*)

   Bit length of the frameID_ field to be put in the header stream of bits.

6. DUMMY_CONTENT_Bits (*DUMMY_ CONTENT_ Bits*)

   Bit length of the Mbit_ field to be put in the header stream of bits.

2.13.2 Commands

1. packerInit
2. printMap
3. printAllFields
4. addPacker

2.14 Module/UW/AL (*Uwal*) : *MPhy*

2.14.1 Bound variables

1. nodeID (*nodeID*)

   Node ID

2. PSDU (*PSDU*)

   size of the PSDU

3. `debug_` (*debug_*)

   Flag to enable debug mode (i.e., printing of debug messages) if set to 1.

4. `interframe_period` (*interframe_period*)

   Time period [s] between two successive frame to be sent down.

5. `frame_set_validity` (*frame_set_validity*)

   Time of validity of a frame set

6. `frame_padding` (*frame_padding*)

   Flag to determine if perfoming bit padding up to PSDU size.

7. `force_endTx` (*force_endTx_*)

   0 not force, otherwise force endTx

2.14.2 Commands

1. `Reset_PER_List`
2. `linkPacker`
3. `setDummyStr`
4. `Set_PER_List`
5. `Clear_PER_List`

2.15 `Module/UW/HMMPHYSICAL` (*UnderwaterHMMPhysical*) : *UnderwaterPhysical*

2.15.1 Bound variables

1. `step_duration` (*step_duration*)

   sampling period for channel transitions

2.15.2 Commands

1. `getPktsTotBad`
2. `getPktsTotGood`
3. `setMCLink`

2.16 Module/UW/HMMPHYSICAL/MCLINK (*MCLink*) : *TclObject*

2.16.1 Bound variables

1. p_succ_good (*p_succ_good*)
   Prob of successful reception with good channel
2. p_succ_bad (*p_succ_bad*)
   Prob of successful reception with bad channel
3. p_gb (*p_gb*)
   Prob of transition from good to bad channel
4. p_bg (*p_bg*)
   Prob of transition from bad to good channel
5. ch_state (*ch_state*)
   last channel state
6. last_step (*last_step*)
   last time step associate to channel state

2.16.2 Commands

1. getLastStep
2. getChState
3. getPSucc

2.17 Module/UW/UwModem/ModemCSA (*UwModemCSA*) : *UwModem {*

2.17.1 Bound variables

1. buffer_size (*DATA_BUFFER_LEN*)
   Size of the buffer that holds data
2. max_read_size (*MAX_READ_BYTES*)
   Maximum number of bytes to be read by a single dump of data

2.17.2 Commands

1. `setServer`
2. `setTCP`
3. `setUDP`

2.18 Module/UW/OPTICAL/PHY (*UwOpticalPhy*) : *MPhy_Bpsk*

2.18.1 Bound variables

1. `Id_` (Id)
2. `Il_` (Il)
3. `R_` (R)
4. `S_` (S)
5. `T_` (T)
6. `Ar_` ($Ar\_$)

2.18.2 Commands

1. `useLUT`
2. `useWOSS`
3. `setVariableTemperature`
4. `setLUTFileName`
5. `setLUTSeparator`

2.19 Module/UW/PHYSICALDB (*UnderwaterPhysicaldb*) : //

2.19.1 Bound variables

2.19.2 Commands

1. `addr`
2. `setCountry`
3. `setModulation`

4. addSnr
5. addSir
6. addOverlap
7. setPath
8. setInterference
9. addRange
10. addTypeOfNode
11. addRangeNum

3 data_link layer

3.1 Module/UW/TLOHI (*MMacTLOHI*) : *MMac*

3.1.1 Bound variables

1. max_prop_delay (*max_prop_delay*)
   One way maximum propagation delay (in seconds) in the network
2. HDR_size (*HDR_size*)
   Size of the HDR if any
3. ACK_size (*ACK_size*)
   Size of the ACK.
4. max_tx_rounds (*max_tx_rounds*)
   Maximum transmission round for one packet
5. wait_costant (*wait_costant*)
   Adding factor in the calculation of the listen time
6. debug_ (*debug_*)
   Debug variable: 0 for no info, >-5 for small info, <-5 for complete info
7. max_payload (*max_payload*)
   Maximum number of payload in a packet.

8. `recontend_time` (*recontend_ time*)
   Time needed for the recontention
9. `tone_data_delay` (*tone_ data_ delay*)
   Not used anymore
10. `max_tx_tries` (*max_ tx_ tries*)
    Maximum number of retransmissions attempt.
11. `buffer_pkts` (*buffer_ pkts*)
    Number of packets a node can store in the container

3.1.2 Commands

1. `addTonePhy`
2. `addDataPhy`
3. `setDataName`
4. `setMacAddr`
5. `setAckMode`
6. `setNoAckMode`
7. `setConservativeUnsyncMode`
8. `setAggressiveUnsyncMode`
9. `setSyncMode`
10. `initialize`
11. `printTransitions`
12. `getCRTime`
13. `getQueueSize`
14. `getTonePktsTx`
15. `getTonePktsRx`
16. `getUpLayersDataRx`

3.2 MInterference/MIV/WKUP (*MInterfMivUwWakeUp*) : //

3.2.1 Bound variables

3.2.2 Commands

3.3 Module/MPhy/Underwater/WKUP (*MPhy_ WakeUp*) : *MPhy*

3.3.1 Bound variables

1. AcquisitionThreshold_dB_ (*AcquisitionThreshold_ dB_*)
   *How many dB over noise are required * for a signal to trigger * acquisition (i.e., a RX attempt)*
2. ToneDuration_ (*ToneDuration_*)
   predefined tone duration
3. MaxTxRange_ (*MaxTxRange_*)
   Maximum Transmission Range

3.3.2 Commands

1. getDroppedPktsTxPending

3.4 Module/UW/ALOHA (*UWAloha*) : *MMac*

3.4.1 Bound variables

1. HDR_size_ (*HDR_ size*)
   Size of the HDR if any
2. ACK_size_ (*ACK_ size*)
   Size of the ACK.
3. max_tx_tries_ (*max_ tx_ tries*)
   Maximum number of retransmissions attempt.
4. wait_constant_ (*wait_ constant*)
   This fixed time is employed to componsate different time variations.
5. uwaloha_debug_ (*uwaloha_ debug*)
   Debuging Flag

6. `max_payload_` (*max_ payload*)

   Maximum number of payload in a packet.

7. `ACK_timeout_` (*ACK_ timeout*)

   ACK timeout for the initial packet

8. `alpha_` (*alpha_*)

   This variable is used to tune the RTT

9. `buffer_pkts_` (*buffer_ pkts*)

   Number of packets a node can store in the container

10. `backoff_tuner_` (*backoff_ tuner*)

    Tunes the backoff duration.

11. `max_backoff_counter_` (*max_ backoff_ counter*)

    Maximum number of backoff it will consider while it increases the backoff exponentially

12. `MAC_addr_` (*addr*)

    Previous mac address

3.4.2 Commands

1. `setAckMode`
2. `setNoAckMode`
3. `initialize`
4. `printTransitions`
5. `getQueueSize`
6. `getUpLayersDataRx`
7. `setMacAddr`

3.5 Module/UW/CSMA_ALOHA/TRIGGER/NODE (*UwCsmaAloha_ Trigger_ NODE*) : *MMac*

3.5.1 Bound variables

1. HDR_size_ (*HDR_ size*)

   Size of the HDR if any

2. debug_ (*debug_*)

   Debug variable: 0 for no info, >-5 for small info, <-5 for complete info

3. max_payload_ (*max_ payload*)

   Maximum number of payload in a packet.

4. buffer_pkts_ (*buffer_ pkts*)

   Number of packets a node can store in the container

5. listen_time_ (*listen_ time*)

   A short channel sensing time

6. tx_timer_duration_ (*tx_ timer_ duration*)

   Duration of the time in which the node is allowed to transmit

3.5.2 Commands

1. initialize

2. getQueueSize

3.6 Module/UW/CSMA_ALOHA/TRIGGER/SINK (*UwCsmaAloha_ Trigger_ SINK*) : *MMac*

3.6.1 Bound variables

1. debug_ (*debug_*)

   Debug variable: 0 for no info, >-5 for small info, <-5 for complete info

2. TRIGGER_size_ (*TRIGGER_ size*)

   Size of the TRIGGER packet

3. `tx_timer_duration_` (*tx_ timer_ duration*)

   Duration of the time in which the node is allowed to transmit

3.6.2 Commands

1. `sinkRun`
2. `getNTriggerSent`

3.7 `Module/UW/TDMA` (***UwTDMA***) : ***MMac***

3.7.1 Bound variables

1. `queue_size_` (*max_ queue_ size*)

   Maximum dimension of Queue

2. `frame_duration` (*frame_ duration*)

   Frame duration

3. `debug_` (*debug_*)

   Debug variable: 0 for no info, >-5 for small info, <-5 for complete info

4. `sea_trial_` (*sea_ trial_*)

   Sea Trial flag: To activate if the protocol is going to be tested at the sea

5. `fair_mode` (*fair_ mode*)

   Fair modality on if 1: then only set tot_ slots and common guard_ time

6. `HDR_size_` (*HDR_ size*)

   Size of the HDR if any

7. `max_packet_per_slot` (*max_ packet_ per_ slot*)

   max numer of packet it can transmit per slot

8. `drop_old_` (*drop_ old_*)

   flag to set the drop packet policy in case of buffer overflow: if 0 (default) drops the new packet, if 1 the oldest

9. `checkPriority_` (*checkPriority*)

   flag to set to 1 if UWCBR module uses packets with priority, set to 0 otherwise. Priority can be used only with UWCBR module

10. `mac2phy_delay_` (*mac2phy_ delay_*)

11. `guard_time` (*guard_ time*)

    A time which is used to componsate variating in timing

12. `tot_slots` (*tot_ slots*)

    Number of slots in the frame (fair_ mode)

3.7.2 Commands

1. `start`
2. `stop`
3. `get_buffer_size`
4. `get_upper_data_pkts_rx`
5. `get_sent_pkts`
6. `get_recv_pkts`
7. `setStartTime`
8. `setSlotDuration`
9. `setGuardTime`
10. `setSlotNumber`
11. `setMacAddr`
12. `setLogLabel`

3.8 `Module/UW/TDMA_FRAME` (***UwTDMA_frame***) : ***UwTDMA***

3.8.1 Bound variables

1. `guard_time` (*guard_ time*)

   A time which is used to componsate variating in timing

3.8.2 Commands

1. `start`
2. `stop`
3. `setSlotNumber`
4. `setTopologyIndex`
5. `setSTopologyFileName`
6. `setTopologySeparator`

3.9 `Module/UW/DACAP` (*MMacDACAP*) : *MMac*

3.9.1 Bound variables

1. `t_min` (*t_ min*)

   Minimum time needed to do an hand-shaking

2. `T_W_min` (*T_ W_ min*)

   Minimum Warning Time in sencods

3. `delta_D` (*delta_ D*)

   Value (in m) that indicates how far we want the CTS propagates over the sender before initiate the data transmission process (it determines the T_ w value)

4. `delta_data` (*delta_ data*)

   Dimension difference (in bytes) among data packets (0 if the packets have always the same dimension)

5. `max_prop_delay` (*max_ prop_ delay*)

   One way maximum propagation delay (in seconds) in the network

6. `CTS_size` (*CTS_ size*)

   Size (in bytes) of the CTS packet

7. `RTS_size` (*RTS_ size*)

   Size (in bytes) of the RTS packet

8. `WRN_size` (*WRN_ size*)

   Size (in bytes) of the WRN packet

9. `HDR_size` (*HDR_ size*)

   Size of the HDR if any

10. `ACK_size` (*ACK_ size*)

    Size of the ACK.

11. `backoff_tuner` (*backoff_ tuner*)

    Tunes the backoff duration.

12. `wait_costant` (*wait_ costant*)

    Adding factor in the calculation of the listen time

13. `debug_` (*debug_*)

    Debug variable: 0 for no info, >-5 for small info, <-5 for complete info

14. `max_payload` (*max_ payload*)

    Maximum number of payload in a packet.

15. `max_tx_tries` (*max_ tx_ tries*)

    Maximum number of retransmissions attempt.

16. `buffer_pkts` (*buffer_ pkts*)

    Number of packets a node can store in the container

17. `alpha_` (*alpha_*)

    This variable is used to tune the RTT

18. `max_backoff_counter` (*max_ backoff_ counter*)

    Maximum number of backoff it will consider while it increases the backoff exponentially

3.9.2 Commands

1. `printTransitions`
2. `setAckMode`
3. `setNoAckMode`
4. `setBackoffFreeze`
5. `setBackoffNoFreeze`
6. `setMultiHopMode`
7. `getQueueSize`
8. `getMeanDeferTime`
9. `getTotalDeferTimes`
10. `getWrnPktsTx`
11. `getWrnPktsRx`
12. `getRtsPktsTx`
13. `getRtsPktsRx`
14. `getCtsPktsTx`
15. `getCtsPktsRx`
16. `getUpLayersDataRx`
17. `setMacAddr`

3.10 `Module/UW/UFETCH/AUV` (***uwUFetch_AUV***) : ***MMac***

3.10.1 Bound variables

1. `T_min_RTS_` (*T_MIN_RTS*)

   Lower bound of the interval in which HN choice the back-off time to tx RTS pck

2. `T_max_RTS_` (*T_MAX_RTS*)

   Upper bound of the interval in which HN choice the back-off time to tx RTS pck

3. `T_guard_` (T_GUARD)

   Guard time interval used between two consecutive transmissions of data packets

4. `t_RTS_` (T_RTS)

   Interval time in which the AUV want to receive an RTS packet in answer to the trigger

5. `MAX_PAYLOAD` ($MAX\_PAYLOAD$)

   Maximum size of DATA PAYLOAD packet

6. `num_max_DATA_AUV_want_receive_` ($NUM\_MAX\_DATA\_AUV\_WANT\_RX$)

   Maximum number of data packet that AUV want to receive from the HN in a single cycle of TRIGGER-RTS-CTS-DATA

7. `TIME_BEFORE_TX_TRIGGER_PCK_` ($T\_START\_PROC\_TRIGGER$)

   Time before that the AUV start the procedure to transmit a TRIGGER packet

8. `MY_DEBUG_` ($debugMio\_$)

   Used if we want to create the logging file

9. `NUMBER_OF_RUN_` (N_RUN)

   Number of run in execution

10. `HEAD_NODE_1_` ($HEAD\_NODE\_1$)

    Id number of HN 1

11. `HEAD_NODE_2_` ($HEAD\_NODE\_2$)

    Id number of HN 2

12. `HEAD_NODE_3_` ($HEAD\_NODE\_3$)

    Id number of HN 3

13. `HEAD_NODE_4_` ($HEAD\_NODE\_4$)

    Id number of HN 4

14. `MODE_COMM_` ($mode\_comm\_hn\_auv$)

    Indicate how the communication takes place with or without RTS-CTS packets

15. `NUM_HN_NETWORK_` (*NUM_HN_NET*)
    Number of Head Nodes in the network

3.10.2 Commands

1. `initialize`
2. `printTransitions`
3. `getTRIGGERtxByAUV`
4. `getRTSrxByAUV`
5. `getRTSCorruptedRxByAUV`
6. `getCTStxByAUV`
7. `getDataRxByAUV`
8. `getDataCorruptedRxByAUV`
9. `AUVNodeStart`
10. `setMacAddr`
11. `initialize`
12. `printTransitions`
13. `getTRIGGERtxByAUV`
14. `getRTSrxByAUV`
15. `getRTSCorruptedRxByAUV`
16. `getCTStxByAUV`
17. `getDataRxByAUV`
18. `getDataCorruptedRxByAUV`
19. `AUVNodeStart`
20. `setMacAddr`

3.11 Module/UW/UFETCH/NODE (*uwUFetch_NODE*) : *MMac*

3.11.1 Bound variables

1. TIME_BEFORE_START_COMU_HN_NODE_ (*T_START_PROCEDURE_HN_NODE*)

   Time within HN is enabled to received a TRIGGER packet from AUV. If in this time the AUV never receive a TRIGGER packet start the communication with the SN

2. MAXIMUM_VALUE_BACKOFF_PROBE_ (*T_MAX_BACKOFF_PROBE*)

   Upper bound timer interval of back-off value used by the SN to choice its back-off time before to transmit a PROBE packet

3. MINIMUM_VALUE_BACKOFF_PROBE_ (*T_MIN_BACKOFF_PROBE*)

   Lower bound timer interval of back-off value used by the SN to choice its back-off time before to transmit a PROBE packet

4. MAXIMUM_NODE_POLLED_ (*MAX_POLLED_NODE*)

   Maximum number of PROBE packets that the HN can receive from the SN after the transmission of a BEACON or CBEACON

5. MAXIMUM_PAYLOAD_SIZE_ (*MAX_PAYLOAD*)

   Maximum size of DATA PAYLOAD packet

6. TIME_TO_WAIT_PROBES_PCK_ (*T_PROBE*)

   alias defined to access the ACK SINK HEADER

7. TIME_TO_WAIT_POLL_PCK_ (*T_POLL*)

   alias defined to access the ACK SINK HEADER

8. TIME_BETWEEN_2_DATA_TX_HN_ (*TIME_BETWEEN_2_TX_DATA_HN_AUV*)

   Interval time used by HN before to transmit the next DATA packet to the AUV

9. TIME_BETWEEN_2_DATA_TX_NODE_ (*TIME_BETWEEN_2_TX_DATA_NODE_HN*)

   Interval time used by the SN before to transmit the next DATA packet to the HN

10. SEE_THE_TRANSITIONS_STATE_ (*PRINT_ TRANSITIONS_ INT*)

    * 0 reason because the SN or HN is passed from a state to another state is not logged in a file*

11. GUARD_INTERVAL_ (*T_ GUARD*)

    Guard time interval used between two consecutive transmissions of data packets

12. MAXIMUM_BUFFER_SIZE_ (*MAXIMUM_ BUFFER_ DATA_ PCK_ NODE*)

    Maximum number of DATA packets that the SN can store in Its queue

13. MAXIMUM_CBEACON_TRANSMISSIONS_ (*MAX_ ALLOWED_ CBEACON_ TX*)

    Interval time in which HN is enabled to received PROBE packets from SNs after the transmission of TRIGGER packet

14. MAXIMUM_PCK_WANT_RX_HN_FROM_NODE_ (*MAX_ PCK_ HN_ WANT_ RX_ FROM_ NODE*)

15. MY_DEBUG_ (*debugMio_*)

    Used if we want to create the logging file

16. NUMBER_OF_RUN_ (*N_ RUN*)

    Number of run in execution

17. TIME_TO_WAIT_CTS_ (*T_ CTS*)

18. MODE_COMM_ (*MODE_ COMM_ HN_ AUV*)

    Indicate the type of communication between HN and AUV, 0 = communication with RTS-CTS, 1 = communication without RTS-CTS

19. BURST_DATA_ (*MODE_ BURST_ DATA*)

    Indicate if it's used or not the burst data. 0=not use burst date, 1=use burst data.

3.11.2 Commands

1. initialize
2. printTransitions

3. getDataQueueSize
4. getBEACONrxByNODE
5. getBEACONrxCorruptedByNODE
6. getPROBEtxByNODE
7. getPOLLrxByNODE
8. getPOLLrxCorruptedByNODE
9. getDATAtxByNODE
10. getCBEACONrxByNODE
11. getCBEACONrxCorruptedByNODE
12. SimpleNodeStart
13. getBEACONtxByHN
14. getPROBErxByHN
15. getPROBErxCorruptedByHN
16. getPOLLtxByHN
17. getDATArxByHN
18. getDATArxCorruptedByHN
19. getCBEACONtxbyHN
20. getTRIGGERrxByHN
21. getTRIGGERrxCorrupteByHN
22. getRTStxByHN
23. getCTSrxByHN
24. getCTSrxCorrupteByHN
25. getDATAtxByHN
26. HeadNodeStart
27. BeHeadNode
28. setMacAddr

3.12 Module/UW/UFETCH/AUV (*uwUFetch_AUV*) : *MMac*

3.12.1 Bound variables

1. T_min_RTS_ (*T_MIN_RTS*)

   Lower bound of the interval in which HN choice the back-off time to tx RTS pck

2. T_max_RTS_ (*T_MAX_RTS*)

   Upper bound of the interval in which HN choice the back-off time to tx RTS pck

3. T_guard_ (*T_GUARD*)

   Guard time interval used between two consecutive transmissions of data packets

4. t_RTS_ (*T_RTS*)

   Interval time in which the AUV want to receive an RTS packet in answer to the trigger

5. MAX_PAYLOAD (*MAX_PAYLOAD*)

   Maximum size of DATA PAYLOAD packet

6. num_max_DATA_AUV_want_receive_ (*NUM_MAX_DATA_AUV_WANT_RX*)

   Maximum number of data packet that AUV want to receive from the HN in a single cycle of TRIGGER-RTS-CTS-DATA

7. TIME_BEFORE_TX_TRIGGER_PCK_ (*T_START_PROC_TRIGGER*)

   Time before that the AUV start the procedure to transmit a TRIGGER packet

8. MY_DEBUG_ (*debugMio_*)

   Used if we want to create the logging file

9. NUMBER_OF_RUN_ (*N_RUN*)

   Number of run in execution

10. HEAD_NODE_1_ (*HEAD_NODE_1*)

    Id number of HN 1

11. `HEAD_NODE_2_` (*HEAD_NODE_2*)
    Id number of HN 2
12. `HEAD_NODE_3_` (*HEAD_NODE_3*)
    Id number of HN 3
13. `HEAD_NODE_4_` (*HEAD_NODE_4*)
    Id number of HN 4
14. `MODE_COMM_` (*mode_comm_hn_auv*)
    Indicate how the communication takes place with or without RTS-CTS packets
15. `NUM_HN_NETWORK_` (*NUM_HN_NET*)
    Number of Head Nodes in the network

3.12.2 Commands

1. `initialize`
2. `printTransitions`
3. `getTRIGGERtxByAUV`
4. `getRTSrxByAUV`
5. `getRTSCorruptedRxByAUV`
6. `getCTStxByAUV`
7. `getDataRxByAUV`
8. `getDataCorruptedRxByAUV`
9. `AUVNodeStart`
10. `setMacAddr`
11. `initialize`
12. `printTransitions`
13. `getTRIGGERtxByAUV`
14. `getRTSrxByAUV`

15. `getRTSCorruptedRxByAUV`
16. `getCTStxByAUV`
17. `getDataRxByAUV`
18. `getDataCorruptedRxByAUV`
19. `AUVNodeStart`
20. `setMacAddr`

3.13 `Module/UW/CSMA_ALOHA` (*CsmaAloha*) : *MMac*

3.13.1 Bound variables

1. `HDR_size_` (*HDR_size*)
   Size of the HDR if any
2. `ACK_size_` (*ACK_size*)
   Size of the ACK.
3. `max_tx_tries_` (*max_tx_tries*)
   Maximum number of retransmissions attempt.
4. `wait_costant_` (*wait_costant*)
   Adding factor in the calculation of the listen time
5. `debug_` (*debug_*)
   Debug variable: 0 for no info, >-5 for small info, <-5 for complete info
6. `max_payload_` (*max_payload*)
   Maximum number of payload in a packet.
7. `ACK_timeout_` (*ACK_timeout*)
   ACK timeout for the initial packet
8. `alpha_` (*alpha_*)
   This variable is used to tune the RTT
9. `backoff_tuner_` (*backoff_tuner*)
   Tunes the backoff duration.

10. `buffer_pkts_` (*buffer_ pkts*)

    Number of packets a node can store in the container

11. `max_backoff_counter_` (*max_ backoff_ counter*)

    Maximum number of backoff it will consider while it increases the backoff exponentially

12. `listen_time_` (*listen_ time*)

    A short channel sensing time

3.13.2 Commands

1. `setAckMode`
2. `setNoAckMode`
3. `initialize`
4. `printTransitions`
5. `getQueueSize`
6. `getUpLayersDataRx`
7. `setMacAddr`

3.14 `Module/UW/MLL` (*UWMllModule*) : *Module*

3.14.1 Bound variables

1. `enable_addr_copy_` (*enable_ addr_ copy*)

3.14.2 Commands

1. `reset`
2. `getArpPacketDrop`
3. `addentry`

3.15 Module/UW/POLLING/NODE (*Uwpolling_ NODE*) : *MMac*

3.15.1 Bound variables

1. T_poll_guard_ (*T_ poll_ guard*)
   Guard time for initial POLL timer
2. backoff_tuner_ (*backoff_ tuner*)
   Tunes the backoff duration.
3. max_payload_ (*max_ payload*)
   Maximum number of payload in a packet.
4. buffer_data_pkts_ (*buffer_ data_ pkts*)
   Length of buffer of DATA pkts in number of pkts
5. Max_DATA_Pkts_TX_ (*max_ data_ pkt_ tx*)
   Max number of DATA packets to transmit each cycle
6. node_id_ (*node_ id*)
   Unique Node ID
7. print_stats_ (*print_ stats*)
   Print protocol's statistics of the protocol
8. sea_trial_ (*sea_ trial*)
   Sea Trial flag: To activate if the protocol is going to be tested at the sea
9. intra_data_guard_time_ (*Intra_ data_ Guard_ Time*)
   Guard Time between one data packet and the following
10. n_run_ (*n_ run*)
    Print protocol's statistics of the protocol
11. useAdaptiveTpoll_ (*useAdaptiveTpoll*)
    True if an adaptive T_ poll is used

3.15.2 Commands

1. `initialize`
2. `getDataQueueSize`
3. `getDataQueueLog`
4. `getProbeSent`
5. `getTimesPolled`
6. `getTriggerReceived`
7. `getTriggerDropped`
8. `getPollDropped`
9. `setMacAddr`

3.16 `Module/UW/POLLING/AUV` (*Uwpolling_AUV*) : *MMac*

3.16.1 Bound variables

1. `max_payload_` (*max_payload*)
   Maximum number of payload in a packet.
2. `T_probe_guard_` (*T_probe_guard*)
   Guard time for PROBE packet: T_probe=T_max+T_probe_guard
3. `T_min_` (*T_min*)
   Minimum value in which the node can choose his backoff time
4. `T_max_` (*T_max*)
   Maximum value in which the node can choose his backoff time
5. `T_guard_` (*T_guard*)
   Guard time added to the calculation of the data TO
6. `T_ack_timer_` (*T_ack_timer*)
   Guard time for PROBE packet: T_probe=T_max+T_probe_guard
7. `max_polled_node_` (*max_polled_node*)
   Maximum number of node that the AUV can poll each time.

8. `sea_trial_` (*sea_ trial_*)

   Sea Trial flag: To activate if the protocol is going to be tested at the sea

9. `print_stats_` (*print_ stats_*)

   Print protocol's statistics of the protocol

10. `modem_data_bit_rate_` (*modem_ data_ bit_ rate*)

    Bit rate of the modem used

11. `n_run_` (*n_ run*)

    Print protocol's statistics of the protocol

12. `Data_Poll_guard_time_` (*DATA_ POLL_ guard_ time_*)

    Guard time between the reception of the last data and the transmission of the following POLL

13. `max_buffer_size_` (*max_ buffer_ size*)

    Max size for the transmission buffer

14. `max_tx_pkts_` (*max_ tx_ pkts*)

    Max number of packets can be transmitted by the AUV during a TxData session

15. `ack_enabled_` (*ack_ enabled*)

    True if ack is enabled, false if disabled, default true

16. `full_knowledge_` (*full_ knowledge*)

    Set to a number != 0 means we have full_ knowledge about the estimate of neighbors

3.16.2 Commands

1. `initialize`
2. `run`
3. `stop_count_time`
4. `GetTotalReceivingTime`
5. `getTriggerSent`

6. `getWrongNodeDataSent`
7. `getProbeReceived`
8. `getPollSent`
9. `getDroppedProbePkts`
10. `getDroppedProbeWrongState`
11. `setMacAddr`
12. `getRxFromNode`
13. `set_adaptive_backoff_LUT`
14. `setLUTSeparator`

3.17 `Module/UW/POLLING/SINK` (*Uwpolling_ SINK*) : *MMac*

3.17.1 Bound variables

1. `T_data_guard_` (*T_ data_ gurad*)
   Guard time for RxDataTimer
2. `backoff_tuner_` (*backoff_ tuner*)
   Tunes the backoff duration.
3. `sink_id_` (*sink_ id*)
   Unique Node ID
4. `sea_trial_` (*sea_ trial*)
   Sea Trial flag: To activate if the protocol is going to be tested at the sea
5. `n_run_` (*n_ run*)
   Print protocol's statistics of the protocol
6. `print_stats_` (*print_ stats*)
   Print protocol's statistics of the protocol
7. `useAdaptiveTdata_` (*useAdaptiveTdata*)
   True if an adaptive T_ poll is used

8. `ack_enabled_` (*ack_ enabled*)

   True if ack is enabled, false if disabled, default true

9. `max_n_ack_` (*max_ n_ ack*)

   Max number of ACK that can be sent in a single round. The same value has to be used in packer, if needed.

10. `T_guard_` (*T_ guard*)

    Guard time added to the calculation of the data TO

11. `max_payload_` (*max_ payload*)

    Maximum number of payload in a packet.

12. `modem_data_bit_rate_` (*modem_ data_ bit_ rate*)

    Bit rate of the modem used

3.17.2 Commands

1. `initialize`
2. `getProbeSent`
3. `getAckSent`
4. `getTriggerReceived`
5. `getTriggerDropped`
6. `getDuplicatedPkts`
7. `setMacAddr`

3.18 `Module/UW/CSMA_CA` (*CsmaCa*) : *MMac*

3.18.1 Bound variables

1. `queue_size_` (*max_ queue_ size*)

   Maximum dimension of Queue

2. `backoff_delta_` (*backoff_ delta*)

   Delta value (configurable) to be added to backoff

3. `backoff_max` (*backoff_max*)
   Maximum value in range of backoff
4. `data_size_` (*data_size*)
   Size of DATA packet
5. `bitrate_` (*bitrate*)
   Bit rate adopted
6. `cts_wait_val_` (*cts_wait_val*)
   Timer duration of CTS
7. `data_wait_val_` (*data_wait_val*)
   Timer duration of DATA
8. `ack_wait_val_` (*ack_wait_val*)
   Timer duration of ACK
9. `log_level_` (*log_level*)
   Current log level chosen for protocol

3.18.2 Commands

1. `initialize`
2. `setAckMode`
3. `setNoAckMode`
4. `getCTSDropped`
5. `getRTSDropped`
6. `getDataDropped`
7. `getQueueSize`
8. `getUpDataRx`
9. `getRTSRx`
10. `getCTSRx`
11. `setMacAddr`

3.19 Module/UW/USR (*MMacUWSR*) : *MMac*

3.19.1 Bound variables

1. HDR_size_ (*HDR_ size*)
   Size of the HDR if any
2. ACK_size_ (*ACK_ size*)
   Size of the ACK.
3. max_tx_tries_ (*max_ tx_ tries*)
   Maximum number of retransmissions attempt.
4. wait_costant_ (*wait_ constant*)
   This fixed time is employed to componsate different time variations.
5. uwsr_debug (*uwsr_ debug*)
   Debuging flag.
6. max_payload_ (*max_ payload*)
   Maximum number of payload in a packet.
7. ACK_timeout_ (*ACK_ timeout*)
   ACK timeout for the initial packet
8. alpha_ (*alpha_*)
   This variable is used to tune the RTT
9. backoff_tuner_ (*backoff_ tuner*)
   Tunes the backoff duration.
10. buffer_pkts_ (*buffer_ pkts*)
    Number of packets a node can store in the container
11. max_backoff_counter_ (*max_ backoff_ counter*)
    Maximum number of backoff it will consider while it increases the backoff exponentially
12. listen_time_ (*listen_ time*)
    A short channel sensing time

13. `guard_time_` (*guard_ time*)
    A time which is used to componsate variating in timing
14. `node_speed_` (*node_ speed*)
    Speed of the mobile node [m/s]
15. `var_k_` (*var_ k*)
    It is employed to decrease the window size.
16. `uwsr_debug_` (*uwsr_ debug*)
    Debuging flag.

3.19.2 Commands

1. `initialize`
2. `printTransitions`
3. `getQueueSize`
4. `getBackoffCount`
5. `getAvgPktsTxIn1RTT`
6. `setMacAddr`

4 network layer

4.1 `Module/UW/SUNNode` (*SunIPRoutingNode*) : *Module*

4.1.1 Bound variables

1. `ipAddr_` (*ipAddr_*)
   IP of the current node.
2. `metrics_` (*metrics_*)
   Metric used by the current node.
3. `PoissonTraffic_` (*PoissonTraffic_*)
   Enable (1) or disable (0) the Poisson traffic for SUN packets.

4. `period_status_` (*period_ status_*)

   Period of the Poisson traffic for status and ack packets.

5. `period_data_` (*period_ data_*)

   Period of the Poisson traffic for data packets in the buffer.

6. `max_ack_error_` (*max_ ack_ error_*)

   Maximum number of Ack errors tollerated by the node.

7. `timer_route_validity_` (*timer_ route_ validity_*)

   Maximum validity time for a route entry.

8. `timer_sink_probe_validity_` (*timer_ sink_ probe_ validity_*)

   Maximum validity time for a sink probe.

9. `timer_buffer_` (*timer_ buffer_*)

   Timer for buffer management.

10. `timer_search_path_` (*timer_ search_ path_*)

    Timer for the search path mechanism.

11. `alpha_` (*alpha_*)

    Parameters used by Load metric. It is a correlation factor.

12. `printDebug_` (*printDebug_*)

    Flag to enable or disable dirrefent levels of debug.

13. `probe_min_snr_` (*probe_ min_ snr_*)

    Value below which if a node receives a probe it discards it.

14. `buffer_max_size_` (*buffer_ max_ size_*)

    Maximum length of the data buffer.

15. `safe_timer_buffer_` (*safe_ timer_ buffer_*)

    Enables a mechanism used to modify the timer_ buffer_ in case of the sending time is shorter than the time needed to receive acks.

16. `disable_path_error_` (*disable_ path_ error_*)

    Flag to enable or disable the possibility to send Path Error packets.

17. `reset_buffer_if_error_` (*reset_ buffer_ if_ error_*)

    If == 1 when a node identify a broken link it will automatically free its buffer.

18. `max_retx_` (*max_ retx_*)

    Maximum Number of transmissions performed: real retransmissions counter the counter is increased only when the packet is sent downlayer

4.1.2 Commands

1. `initialize`
2. `clearhops`
3. `printhopcount`
4. `printhops`
5. `printselectedroutes`
6. `getackcount`
7. `getdatapktcount`
8. `getforwardedcount`
9. `getdatapktdroppedbuffer`
10. `getdatapktdroppedmaxretx`
11. `getpathestablishmentpktcount`
12. `getackheadersize`
13. `getdatapktheadersize`
14. `getpathestheadersize`
15. `getNpathsestablished`
16. `getbufferstatus`
17. `getmeanretx`
18. `gettransmittedpackets`

19. `getstats`
20. `addr`
21. `trace`

4.2 `Module/UW/SUNSink` (*SunIPRoutingSink*) : *Module*

4.2.1 Bound variables

1. `t_probe` (*t_probe*)
   Period of the probing.
2. `ipAddr_` (*ipAddr_*)
   IP of the current node.
3. `PoissonTraffic_` (*PoissonTraffic_*)
   Enable (1) or disable (0) the Poisson traffic for SUN packets.
4. `periodPoissonTraffic_` (*periodPoissonTraffic_*)
   Period of the Poisson traffic.
5. `printDebug_` (*printDebug_*)
   Flag to enable or disable dirrefent levels of debug.

4.2.2 Commands

1. `initialize`
2. `start`
3. `stop`
4. `sendprobe`
5. `getprobetimer`
6. `getprobepktcount`
7. `getackcount`
8. `getprobepktheadersize`

9. `getackheadersize`
10. `setnumberofnodes`
11. `addr`
12. `trace`
13. `tracepaths`
14. `getstats`

4.3 `Module/UW/PosBasedRt` (*UwPosBasedRt*) : *Module*

4.3.1 Bound variables

1. `debug_` (*debug_*)
   Flag to enable or disable dirrefent levels of debug.
2. `maxTxRange_` (*maxTxRange*)
   Maximum transmission range, in meters, for this node.
3. `ROV_speed_` (*ROV_speed*)
   Last known ROV speed.

4.3.2 Commands

1. `setMaxTxRange`
2. `addr`
3. `setNodePosition`
4. `addRoute`
5. `toMovingNode`
6. `toFixedNode`

4.4 Module/UW/PosBasedRt/ROV (*UwPosBasedRtROV*) : *Module*

4.4.1 Bound variables

1. debug_ (*debug_*)

   Flag to enable or disable dirrefent levels of debug.

2. maxTxRange_ (*maxTxRange*)

   Maximum transmission range, in meters, for this node.

4.4.2 Commands

1. setMaxTxRange
2. addr
3. setROVPosition
4. addPosition_IPotherNodes

4.5 Module/UW/IP (*UWIPModule*) : *Module*

4.5.1 Bound variables

1. debug_ (*debug_*)

   Flag to enable or disable dirrefent levels of debug.

4.5.2 Commands

1. addr
2. setaddrinet
3. setaddrilink
4. addr-string
5. getipheadersize
6. printidspkts
7. addr

4.6 Module/UW/FLOODING (*UwFlooding*) : *Module*

4.6.1 Bound variables

1. ttl_ (*ttl_*)

   Time to leave of the packet.

2. maximum_cache_time_ (*maximum_ cache_ time_*)

   Validity time of a packet entry.

3. optimize_ (*optimize_*)

   Flag used to enable the mechanism to drop packets processed twice.

4. forward_timeout_ (*fwd_ to*)

   Time out within which the forwarding is expected.

5. alpha_snr_ (*alpha_ snr*)

   Value to be used by the NeighborReputationHandler object to combine new snr values and average snr.

4.6.2 Commands

1. getpacketsforwarded
2. getfloodingheadersize
3. printNeighbor
4. addr
5. trace
6. setReputation
7. setPhyTag
8. addTtlPerTraffic

4.7 Module/UW/ICRPNode (*UwIcrpNode*) : *Module*

4.7.1 Bound variables

1. printDebug_ (*printDebug_*)
   Flag to enable or disable dirrefent levels of debug.
2. maxvalidititytime_ (*max_ validity_ time_*)
   Maximum validity time of a route.
3. timer_ack_waiting_ (*timer_ ack_ waiting_*)
   Ack waiting timer.

4.7.2 Commands

1. initialize
2. clearhops
3. printhops
4. getackheadersize
5. getdataheadersize
6. getstatusheadersize
7. getackpktcount
8. getdatapktcount
9. getstatuspktcount
10. ipsink
11. addr

4.8 Module/UW/ICRPSink (*UwIcrpSink*) : *Module*

4.8.1 Bound variables

1. printDebug_ (*printDebug_*)
   Flag to enable or disable dirrefent levels of debug.

4.8.2 Commands

1. `initialize`
2. `getackheadersize`
3. `getdataheadersize`
4. `getstatusheadersize`
5. `getackpktcount`
6. `getstatuspktcount`
7. `addr`

4.9 `Module/UW/StaticRouting` (*UwStaticRoutingModule*) : //

4.9.1 Bound variables

4.9.2 Commands

1. `numroutes`
2. `clearroutes`
3. `defaultGateway`
4. `addroute`

5 transport layer

5.1 `Module/UW/UDP` (*UwUdp*) : *Module*

5.1.1 Bound variables

1. `drop_duplicated_packets_` (*drop_ duplicated_ packets_*)
   Flat to enable or disable the drop of duplicated packets.
2. `debug_` (*debug_*)
   Flag to enable or disable dirrefent levels of debug.

5.1.2 Commands

1. `getudpheadersize`
2. `printidspkts`
3. `assignPort`

6 application layer

6.1 `Module/UW/APPLICATION` (*uwApplicationModule*) : *Module*

6.1.1 Bound variables

1. `debug_` (*debug_*)

   Flag to enable or disable dirrefent levels of debug.

2. `period_` (*PERIOD*)

   Interval time between two successive generation data packets

3. `node_ID_` (*node_id*)

   Variable that handle the file in which the protocol write the statistics

4. `EXP_ID_` (*exp_id*)

   Variable that handle the file in which the protocol write the statistics

5. `PoissonTraffic_` (*poisson_traffic*)

   Enable or not the Poisson process for generation of data packets 1 enabled 0 not enabled

6. `Payload_size_` (*payloadsize*)

   Size of each data packet payaload generated ** `destAddr_` (*dst_addr*)

   IP destination address.

7. `destPort_` (*port_num*)

   Number of the port in which the server provide the service

8. `Socket_Port_` (*servPort*)

   Server port

9. `drop_out_of_order_` (*drop_ out_ of_ order*)

   Enable or not the ordering of data packet received 1 enabled 0 not enabled

10. `max_read_length` (*MAX_ READ_ LEN*)

    Maximum size (bytes) of a single read of the socket

6.1.2 Commands

1. `start`
2. `stop`
3. `getsentpkts`
4. `lostpkts`
5. `getrecvpkts`
6. `outofsequencepkts`
7. `notknownpktrx`
8. `getrecvpktsqueue`
9. `getrtt`
10. `getrttstd`
11. `getftt`
12. `getfttstd`
13. `getper`
14. `getthr`
15. `print_log`
16. `SetSocketProtocol`
17. `UDP`
18. `TCP`

6.2 Module/UW/CBR (*UwCbrModule*) : *Module*

6.2.1 Bound variables

1. period_ (*period_*)

   Period between two consecutive packet transmissions.

2. destPort_ (*dstPort_*)

   Destination port.

3. destAddr_ (*dstAddr_*)

   IP of the destination.

4. packetSize_ (*pktSize_*)

   Packet size.

5. PoissonTraffic_ (*PoissonTraffic_*)

   1 if the traffic is generated according to a poissonian distribution.

6. debug_ (*debug_*)

   Flag to enable or disable dirrefent levels of debug.

7. drop_out_of_order_ (*drop_out_of_order_*)

   Flag to enable or disable the check for out of order packets.

8. traffic_type_ (*traffic_type_*)

   Traffic type of the packets.

9. tracefile_enabler_ (*tracefile_enabler_*)

   True if enable tracefile of received packets, default disabled.

6.2.2 Commands

1. start
2. stop
3. getrtt
4. getftt
5. getper

6. `getthr`
7. `getcbrheadersize`
8. `getrttstd`
9. `getfttstd`
10. `getsentpkts`
11. `getrecvpkts`
12. `setprioritylow`
13. `setpriorityhigh`
14. `sendPkt`
15. `sendPktLowPriority`
16. `sendPktHighPriority`
17. `resetStats`
18. `printidspkts`
19. `setLogSuffix`
20. `setLogSuffix`

6.3 `Module/UW/VBR` (*UwVbrModule*) : *Module*

6.3.1 Bound variables

1. `period1_` (*period1_*)
   period between two consecutive packet transmissions (mode 1).
2. `period2_` (*period2_*)
   period between two consecutive packet transmissions (mode 2).
3. `timer_switch_1_` (*timer_ switch_ 1_*)
   Period in witch the node tramsmits with a packet every period1_ seconds.

4. `timer_switch_2_` (*timer_ switch_ 2_*)

   Period in witch the node tramsmits with a packet every period2_ seconds.

5. `destPort_` (*dstPort_*)

   Destination port.

6. `destAddr_` (*dstAddr_*)

   IP of the destination.

7. `packetSize_` (*pktSize_*)

   Packet size.

8. `PoissonTraffic_` (*PoissonTraffic_*)

   1 if the traffic is generated according to a poissonian distribution.

9. `debug_` (*debug_*)

   Flag to enable or disable dirrefent levels of debug.

10. `drop_out_of_order_` (*drop_ out_ of_ order_*)

    Flag to enable or disable the check for out of order packets.

6.3.2 Commands

1. `start`
2. `stop`
3. `getrtt`
4. `getftt`
5. `getper`
6. `getthr`
7. `getvbrheadersize`
8. `getrttstd`
9. `getfttstd`

10. `getsentpkts`
11. `getrecvpkts`
12. `sendPkt`
13. `resetStats`

7 mobility layer

7.1 `Position/UWSM` (*UWSMPosition*) : *Position*

7.1.1 Bound variables

1. `debug_` (*debug_*)

   Flag to enable or disable dirrefent levels of debug.

7.1.2 Commands

1. `setdest`
2. `setdest`
3. `update`

7.2 `Position/UWGM` (*UwGMPosition*) : *Position*

7.2.1 Bound variables

1. `xFieldWidth_` (*xFieldWidth_*)

   Range of the x-axis of the field to be simulated, in meters.

2. `yFieldWidth_` (*yFieldWidth_*)

   Range of the y-axis of the field to be simulated, in meters.

3. `zFieldWidth_` (*zFieldWidth_*)

   Range of the z-axis of the field to be simulated, in meters.

4. `alpha_` (*alpha_*)

   Parameter to be used to vary the randomness: 0: totally random values (Brownian motion), 1: linear motion.

5. `alphaPitch_` (*alphaPitch_*)
   Pitch of alpha variable.
6. `updateTime_` (*updateTime_*)
   Time between two update computation.
7. `directionMean_` (*directionMean_*)
   Defines the mean value of the direction.
8. `pitchMean_` (*pitchMean_*)
   Mean value for the pitch.
9. `debug_` (*debug_*)
   Flag to enable or disable dirrefent levels of debug.

7.2.2 Commands

1. `bound`
2. `SPHERIC`
3. `THOROIDAL`
4. `HARDWALL`
5. `REBOUNCE`
6. `speedMean`

7.3 `Position/UWDRIFT` (*UwDriftPosition*) : *Position*

7.3.1 Bound variables

1. `xFieldWidth_` (*xFieldWidth_*)
   Range of the x-axis of the field to be simulated, in meters.
2. `yFieldWidth_` (*yFieldWidth_*)
   Range of the y-axis of the field to be simulated, in meters.
3. `zFieldWidth_` (*zFieldWidth_*)
   Range of the z-axis of the field to be simulated, in meters.

4. `boundx_` (*boundx_*)

   1 if the x-axis is bounded, 0 otherwise.

5. `boundy_` (*boundy_*)

   1 if the y-axis is bounded, 0 otherwise.

6. `boundz_` (*boundz_*)

   1 if the z-axis is bounded, 0 otherwise.

7. `speed_horizontal_` (*speed_horizontal_*)

   Speed of the node in the x-axis, in m/s.

8. `speed_longitudinal_` (*speed_longitudinal_*)

   Speed of the node in the y-axis, in m/s.

9. `speed_vertical_` (*speed_vertical_*)

   Speed of the node in the z-axis, in m/s.

10. `alpha_` (*alpha_*)

    Parameter to be used to vary the randomness: 0: totally random values (Brownian motion), 1: linear motion.

11. `deltax_` (*deltax_*)

    Max value of the Uniform Distribution: Random movement between [0, deltax_).

12. `deltay_` (*deltay_*)

    Max value of the Uniform Distribution: Random movement between [0, deltay_).

13. `deltaz_` (*deltaz_*)

    Max value of the Uniform Distribution: Random movement between [0, deltaz_).

14. `starting_speed_x_` (*starting_speed_x_*)

    Initial speed of the node. x axis in m/s.

15. `starting_speed_y_` (*starting_speed_y_*)

    Initial speed of the node. y axis in m/s.

16. `starting_speed_z_` (*starting_speed_z_*)
    Initial speed of the node. z axis in m/s.
17. `updateTime_` (*updateTime_*)
    Time between two update computation.
18. `debug_` (*debug_*)
    Flag to enable or disable dirrefent levels of debug.

7.3.2 Commands

8 propagation layer

8.1 `Module/UW/OPTICAL/Propagation` (***UwOpticalMPropagation***) : ***MPropagation***

8.1.1 Bound variables

1. `Ar_` (*Ar_*)
   Receiver area [m^2]
2. `At_` (*At_*)
   Transmitter size [m^2]
3. `c_` (*c_*)
   Lookup table map of the attenuation coefficient and the temperature versus the depth
4. `theta_` (*theta_*)
   Transmitting beam diverge angle [rad]
5. `debug_` (*debug_*)

8.1.2 Commands

1. `setOmnidirectional`
2. `setDirectional`
3. `setVariableC`
4. `setFixedC`

5. `setLUT`
6. `setAr`
7. `setAt`
8. `setC`
9. `setTheta`
10. `setLUTFileName`
11. `setLUTSeparator`

9 channel layer

9.1 `Module/UW/Optical/Channel` (*UwOpticalChannel*) : *ChannelModule*

9.1.1 Bound variables

1. `RefractiveIndex_` (*refractive_index*)

   refractive index of the underwater medium.

9.1.2 Commands

10 interference layer

10.1 `Module/UW/INTERFERENCE` (*uwinterference*) : *MInterferenceMIV*

10.1.1 Bound variables

1. `use_maxinterval_` (*use_maxinterval_*)

   set to 1 to use maxinterval_.

10.1.2 Commands