CHAPTER-4

DESIGN AND ANALYSIS OF UNDERWATER WSN

4.1 Introduction

In this chapter, we will study the characteristics of optical and acoustic links and their performance in a wireless sensor network set up in underwater environment in QualNet 5.0 simulator. A sample network constructed on Qualnet acts as a scenario to test custom physical layer protocols and communication medium protocols which have been modelled in chapter-4. This is then used to study the underwater communication and analyze it on the basis of various performance parameters.

4.2 Advantages of a Hybrid Network

Both the optical and acoustic communication have their own pros and cons, therefore hybrid link configurations are the latest area of research. Generally, two type of configuration are used for this network. The first configuration utilizes both acoustic wave and optical wave as duplex transmission medium. In this configuration, the two ends of the link are the sensor nodes that are equipped with both acoustic and optical transceivers. When two nodes of the link are in short distance and water condition is clear, the system will use optical wave as carrier to achieve high speed data transmission. If there is large distance between the two nodes or the water is turbid, the system will instead employ acoustic methods to accomplish connectivity. The virtue of this implementation is the high flexibility and reliability, but at the expense of high power consumption and bulky instruments due to acoustic transceivers on both ends. Moreover, if the number of sensor nodes are large then it is not feasible to make the nodes bulky.

In the second configuration, the system is configured by one static control platform and several mobile sensor nodes. Acoustic wave is used as a broadcasting method to transmit control information in the downlink from control platform to each sensor node. While optical wave is applied in the communication links between each sensor node, as well as uplinks from sensor nodes to main control platform. This hybrid UWC system utilizes the advantages of each communication method. Since acoustic wave has diffusion property and long propagation distance, it can cover the area that is distributed with sensor nodes. Moreover, in the downlink from control platform to sensor nodes, the transmitted information are low-speed control signals, which are suitable for acoustic communication. On the other hand, in the uplink of the system the large volume of oceanic monitoring data is transmitted through high UOWC links.

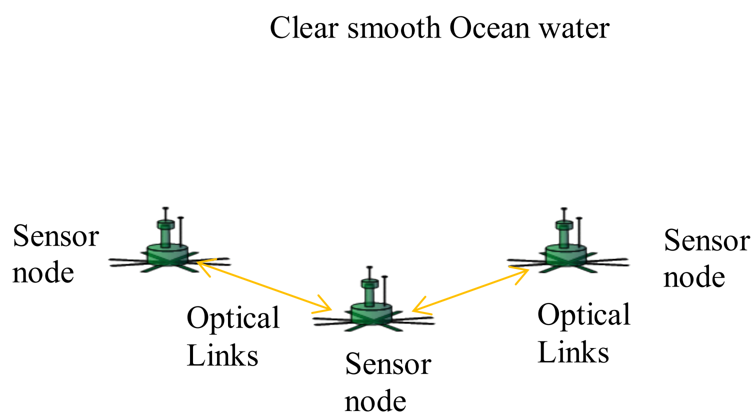

Fig 4.1 Optical Communication in clear ocean waters.

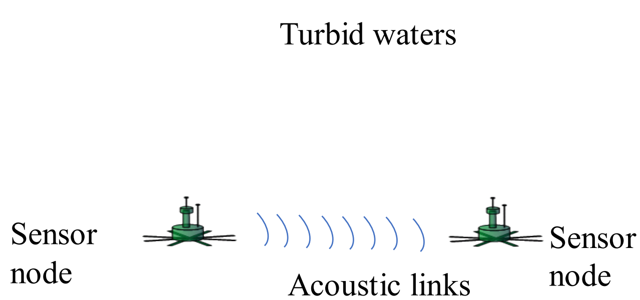

Fig 4.2 Acoustic Communication in turbulent ocean waters.

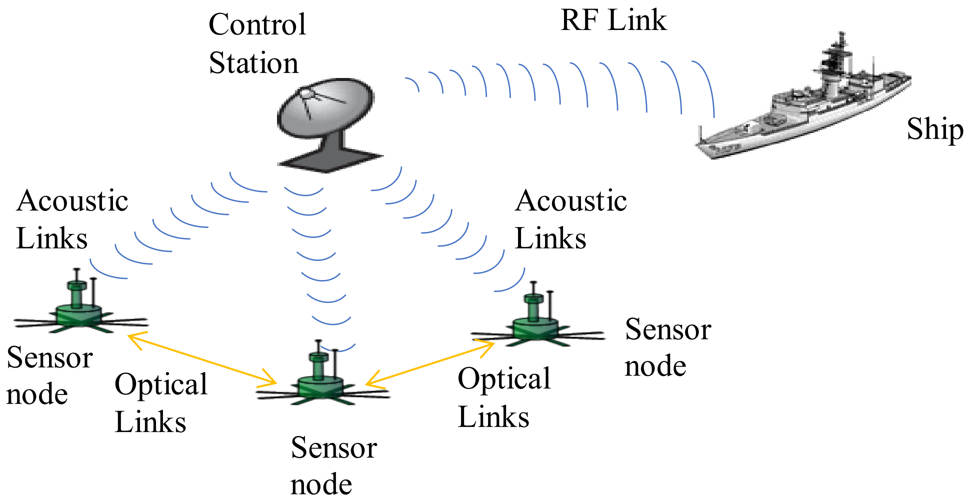

Fig 4.3 Hybrid Opto-Acoustic Sensor Network (Configuration)

4.3 Qualnet Simulation Environment

QualNet is a comprehensive suite of tools for modeling large wired and wireless networks. It uses simulation to predict the behavior and performance of networks to improve their design, operation and management.

QualNet enables users to:

- Design new protocol models.
- Optimize new and existing models.
- Design large wired and wireless networks using pre-configured or user-designed models.
- Analyze the performance of networks and perform what-if analysis to optimize them.

4.3.1 Features of Qualnet

The key features of QualNet that enable creating a virtual network environment are:

- **Speed:** QualNet can support real-time speed to enable software-in-the-loop, network emulation, and hardware in- the-loop modeling. Faster speed enables model developers and network designers to run multiple "what-if" analyses by varying model, network, and traffic parameters in a short time.

- **Scalability:** QualNet can model thousands of nodes by taking advantage of the latest hardware and parallel computing techniques. QualNet can run on cluster, multi-core, and multi-processor systems to model large networks with high fidelity.
- **Model Fidelity:** QualNet uses highly detailed standards-based implementation of protocol models. It also includes advanced models for the wireless environment to enable more accurate modeling of real-world networks.
- **Portability:** QualNet and its library of models run on a vast array of platforms, including Windows, Linux, and Mac OS X operating systems, distributed and cluster parallel architectures, and both 32- and 64-bit computing platforms. Users can now develop a protocol model or design a network in QualNet on their desktop or laptop Windows computer and then transfer it to a powerful multi-processor Linux server to run capacity, performance, and scalability analyses.
- **Extensibility:** QualNet can connect to other hardware and software applications, such as OTB, real networks, and third-party visualization software, to greatly enhancing the value of the network model.

4.3.2 Architecture of Qualnet

The simulator consists of various modules which together constitute of the integrated development environment. Qualnet has scenario development module, its visualizer and an analyzer to analyze the results. The detailed description of the modules shown in Fig.4.4 is given below:

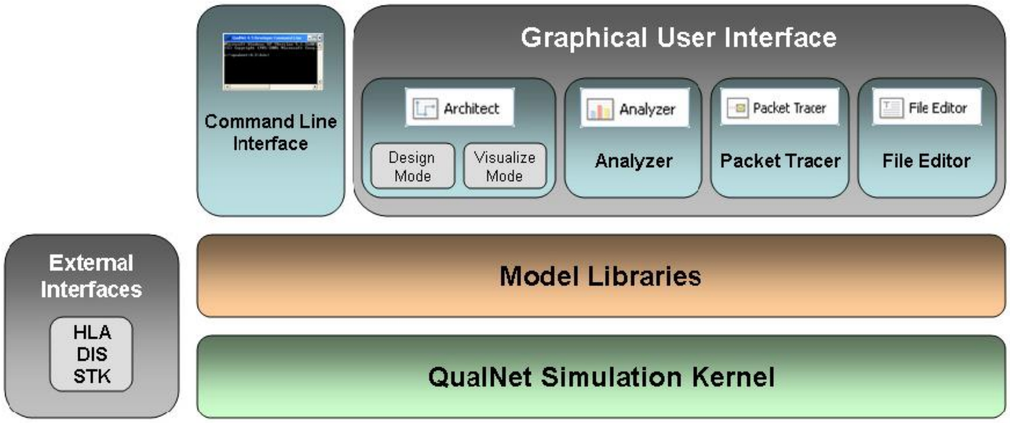

Fig 4.4 Architecture of Qualnet

- **QualNet Kernel:** The kernel of QualNet is a, Scalable Network Technologies-proprietary, parallel discrete-event scheduler. It provides the scalability and portability to run hundreds and thousands of nodes with high-fidelity models on a variety of platforms, from laptops and desktops to high performance computing systems. Users do not directly interact with the kernel, but use the QualNet API to develop their protocol models.

- **QualNet Model Libraries:** QualNet includes support for a number of model libraries that enable you to design networks using protocol models developed by Scalable Network Technologies. Purchase of QualNet includes the Developer, Wireless, and Multimedia and Enterprise Model Libraries; additional libraries for modeling WiMAX, network security, sensor networks, satellite, and cellular models are also available.

- **QualNet Graphical User Interface (GUI):** QualNet GUI consists of Architect, Analyzer, Packet Tracer, and File Editor. These modes are accessible from the Components Toolbar. Architect is a network design and visualization tool. It has two modes: Design mode and Visualize mode.

i. In Design mode, you can set up terrain, network connections, subnets, mobility patterns of wireless users, and other functional parameters of network nodes. You can create network models by using intuitive, click and drag operations. You can also customize the protocol stack of any of the nodes. You can also specify the application layer traffic and services that run on the network.

ii. In Visualize mode, you can perform in-depth visualization and analysis of a network scenario designed in Design mode. As simulations are running, users can watch packets at various layers flow through the network and view dynamic graphs of critical performance metrics. Real-time statistics are also an option, where you can view dynamic graphs while a network scenario simulation is running. You can also assign jobs to run in batch mode on a faster server and view the animated data later. You can perform “what-if” analysis by setting a range of values for a protocol parameter and comparing the network performance results for each of them.

iii. Analyzer is a statistical graphing tool that displays the metrics collected during the simulation of a network scenario in a graphical format. You can customize the graph display. All statistics are exportable to spreadsheets in CSV format.

iv. Packet Tracer provides a visual representation of packet trace files generated during the simulation of a network scenario. Trace files are text files in XML format that contain information about packets as they move up and down the protocol stack.

v. File Editor is a text editing tool that displays the contents of the selected file in text format and allows the user to edit files.

- **QualNet Command Line Interface:** The QualNet command line interface enables a user to run QualNet from a DOS prompt (in Windows) or from a command window (in Linux or Mac OS X). When QualNet is run from the command line, input to QualNet is in the form of text files which can be created and modified using any text editor. Building and running scenarios with the command line interface takes less memory and scenarios typically run faster than with the GUI. With the command line interface the users have the flexibility to interface with visualization and analysis tools of their choice.

- **QualNet External Interfaces:** QualNet can also interact with a number of external tools in real-time. The HLA/DIS module, which is a part of the Standard Interfaces Model Library, allows QualNet to interact with other HLA/DIS compliant simulators and computer-generated force (CGF) tools. The QualNet STK interface, which is a part of the

Developer Model Library, provides a way to interface QualNet with the Satellite Toolkit (STK) developed by Analytical Graphics, Inc. and function in a client-server environment.

4.4 Design of a Communication Medium in QualNet

The communication between nodes happen through the communication medium. QualNet simulates the propagation of signals between nodes, considering both propagation delays and signal attenuation due to path loss, fading and shadowing. Pathloss models are given according to model library being worked upon. A few of them are Two Ray, Free Space, Irregular Terrain Model and Pathloss Matrix. Along with this shadowing model pre-defined in QualNet are Constant and Lognormal. Lastly the fading models that describe a communication channel are Ricean fading based with Rayleigh being a part of it. Modifications and newer developments are allowable for the case of communication channels also. Therefore, there is a need to study that how a channel can be modelled which can behave as a underwater environment. Modelling is done uniquely for every type of communication link i.e Optical and acoustic.

While creating a custom Underwater Communication medium in Qualnet 5.0, we consider the following three models which characterize any medium of communication. A wireless communication medium model in QualNet simulates the propagation of signals between nodes, considering both propagation delays and signal attenuation due to path loss, fading, and shadowing.

- Pathloss Models: From the given pathloss models in wireless library, namely Two Ray model, Pathloss Matrix, ITM and Free Space, none is suitable for our need of underwater scenario. Therefore, custom fading model becomes a necessity.
- Fading Models: Out of given options of Ricean, Rayleigh and Fast Rayleigh, Ricean fading model can be chosen for wireless optical LOS communication as it is meant for scenarios where there is line of sight communication and the line of sight signal is the dominant signal seen at the receiver.
- Shadowing Models: A shadowing model is used to represent the signal attenuation caused by obstructions along the propagation path. The constant shadowing model is suitable for the scenarios without mobility where the obstructions along the propagation paths remain

unchanged. Since we assume that there is no hindrance in the communication, therefore we select ‘None’ as the shadowing model.

4.4.1 Creating custom Communication Channel

The following section gives the actions that need to be performed for adding a path loss model, ‘UWOPathloss’ (Underwater Optical Pathloss Model)/ ‘UWAPathloss’ (Underwater Acoustic Pathloss Model), to QualNet.

Step 1. Create a directory in qualnet /libraries called user_models.

Step 2. Create a subdirectory in qualnet / libraries / user_models / src

Step 3. Create two files prop_UWOP.h/ prop_UWAP.h & prop_UWOP.cpp/prop_UWAP.cpp in that directory.

Step4. The prop_UWOP.h/prop_UWAP.h should contain the two major things, first prototypes for interface functions in the source file, prop_UWOP.cpp/prop_UWAP.cpp and second all the constant definitions.

Step 5. Include the following in the prop_UWOP.cpp/prop_UWAP.cpp

- Statement to include the path loss model’s header file.
- Statements to include standard library functions and other header files needed by the path loss model’s source file.
- Initialization function for the path loss model.
- Path loss calculation function for the path loss model.

Step 6: Add the UWOP/UWAP path loss model to the enumeration PathlossModel defined in the file: qualnet/include/propagation.h

Step 7: Call “UWOPInitialize/ UWAPInitialize” from the function “PROP_GlobalInit” in the code file “qualnet/libraries/wireless/src/propagation.cpp” .

Step 8: Modify the function “PROP_CalculatePathloss” in “qualnet/libraries/wireless/src/propagation.cpp” to call the model’s path loss calculation function. This is done to add UWOP/UWAP path loss model to QualNet.

Step 9: In the directory QUALNET/libraries/ user_models, create a file Makefile-common to include and complile QualNet with UWOP/UWAP path loss model.

Step 10: Create a file Makefile-windows in QUALNET/libraries/user_models/ and include the Makefile-common file into it.

Step 11: To enable the library, make the following entry in the file QUALNET/main/Makefile-addons-windows.

Step 12: After creating and modifying the Makefiles, recompile QualNet. Before recompiling, we must delete all object files to correctly integrate our custom changes.

Step 13: To incorporate the UWOP/ UWAP propagation protocol to the QualNet GUI, the description file, “channel_properties.prt” in the directory “qualnet/gui/settings/protocol models/”, is modified as follows.

Step 14: Restart QualNet 5.0 to see the changes appear in the GUI; Scenario Properties−>Channel Properties−>Pathloss Model.

These steps allow us to create our custom channel in Qualnet. Fig. 4.5 shows the Channel Properties of the network scenarios and the specifications shown correspond to the optical channel ans similarly Fig. 4.6 shows the Channel Properties of the network scenarios and the specifications shown correspond to the acoustic channel.

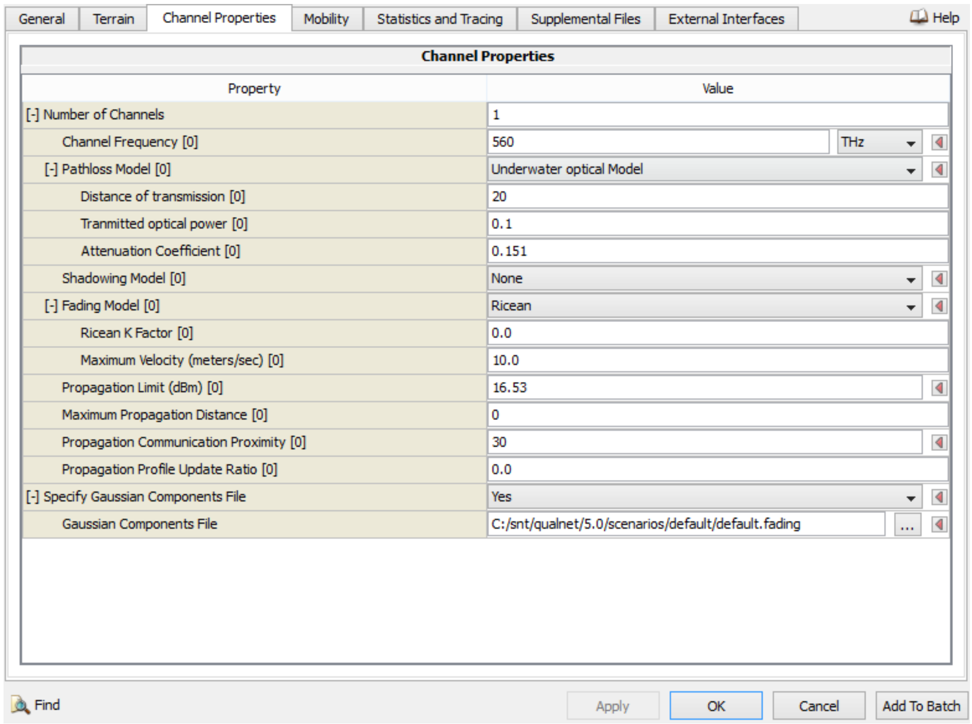

Fig 4.5 Screenshot of Underwater Optical Channel Properties

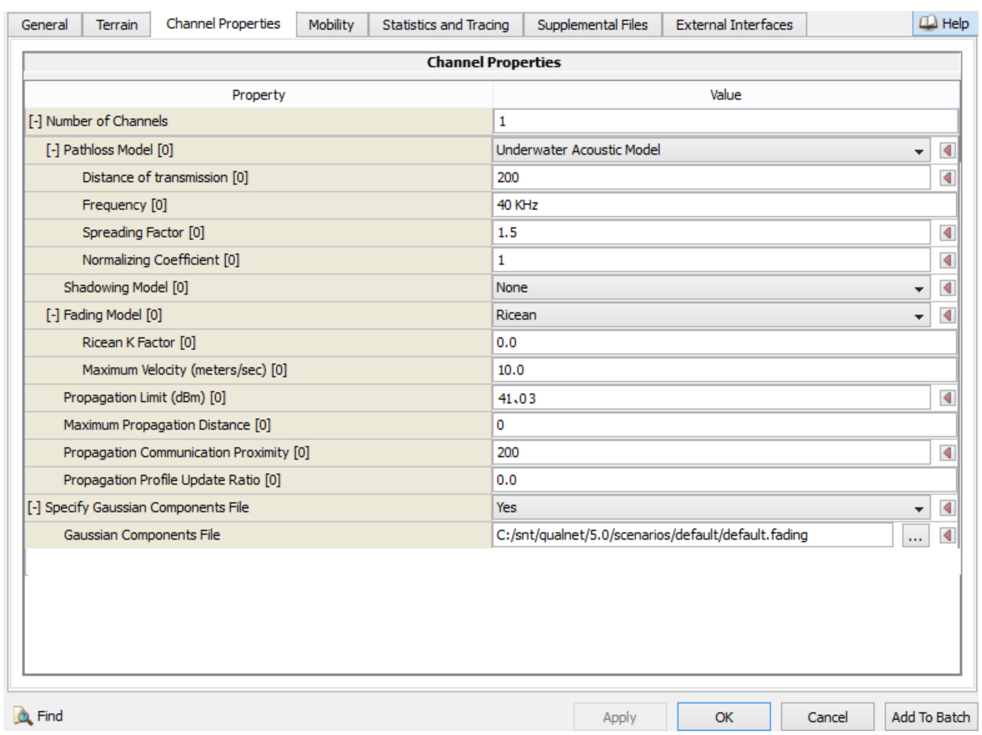

Fig 4.6 Screenshot of Underwater Acoustic Channel Properties

4.5 Creating Environmental Noise Model

The ocean offers various sources of noise which interfere with the signal travelling and thereby in the communication. It is to be studied so that such systems and network can be designed which have significant SNR value for efficient communication.

4.5.1 Optical Environmental Noise Model

The implementation of the underwater equivalent noise power computation is to be done in "PHY_CreateAPhyForMac" function of the source file "qualnet/libraries/wireless/src/phy.cpp". The noise power is computed in a communication node when it is initialized.

Step 1: Add the UWOP model files in the physical layer file phy.cpp

Step 2: Introduce UWOP physical model in the function.

Step 3: Calculate the noise equivalent power as given in Equation 3.10. Using the equations 3.11 to 3.14, modify the function "PHY_CreateAPhyForMac" for the same.
Step 3.a: Declare all parameters.
Step 3.b: Read all input parameters.
Step 3.c: Calculate all expressions of Powers.

4.5.2 Acoustic Environmental Noise Model

The implementation of the underwater equivalent noise power computation is to be done in "PHY_CreateAPhyForMac" function of the source file "qualnet/libraries/wireless/ src/phy.cpp". The noise power is computed in a communication node when it is initialized.

Step 1: Add the UWAP model files in the physical layer file phy.cpp

Step 2: Introduce UWAP physical model in the function.

Step 3: Calculate the total noise power as given in Equation 3.7. Using the equations 3.3 to 3.6. Modify the function PHY_CreateAPhyForMac for the same.

Step 3.a: Declare all parameters.

Step 3.b: Read all input parameters.

Step 3.c: Calculate all expressions of Powers.

4.6 Physical layer modelling for Optical communication

The Physical Layer is the lowest layer in the QualNet protocol stack. It communicates data over physical media. In QualNet, a Physical Layer model consists of two parts:

- PHY component: It models signal transmission and reception and reflects the effects of the MAC scheme, node status, physical parameters, distortions from the channel, and interference from neighbor nodes.
- Antenna component: The antenna component models the functions and properties of the antenna.

  Following are the actions that need to be performed for adding a PHY model, PHY_UWOP, to QualNet.

Step 1: Create header and source files namely phy_UWOP.h and phy_UWOP.cpp. Place them in the folder QUALNET/libraries/user_models/src.

Step 2: The header file, phy_UWOP.h, should contain the following:

- Constant definitions
- Data structure definitions
- Prototypes for interface functions in the source file, phy_UWOP.cpp

Step 3: Specify the Physical model created, PHY_UWOP, in the enumeration in QUALNET/include/phy.h.

Step 4: Each PHY model has its own data structures, which are defined in the model's header file. They store information such as:

- PHY parameters
- Statistics variable

Step 5. For a wireless MAC protocol, function AddNodeToSubnet initializes the Physical Layer model, specified for the interface, by calling the function PHY_CreateAPhyForMac. Modify AddNodeToSubnet to call PHY_CreateAPhyForMac with PHY_UWOP as the PhyModel parameter. This has to be done in QUALNET/main/mac.cpp.

Step 6: Modify the function PHY_CreateAPhyForMac() in phy.cpp to initialize the UWOP physical model using PhyUWOPInit() function. We are not developing any user specified receiver model, rather we are using SNR based receiver model. So any changes pertaining to that are not required.

Step 7: The initialization of a PHY model takes place in the initialization function of the model that is called by the Physical Layer initialization function PHY_CreateAPhyForMac. The initialization function of a PHY model commonly performs the following tasks:

- Create an instance of the PHY model data structure
- Read and store the PHY model's parameters
- Initialize the state variables of the PHY model
- Initialize the antenna model
- Set the transmission channel

Create the PhyUWOPInit() function as follows:

Step 7.a: Declare Parameters
Step 7.b: Get user inputs to calculate the SNR value.
Step 7.c: Calculate the value of SNR
Step 7.d: Set parameters
Step 7.e: Calculate the Bit Rate
Step 7.f: Other APIs code (Similar to Phy802_11Init())

Step 8: Modify the event handler function PhyUWOPTransmissionEnd in phy.cpp which deals with the end of transmission of a packet by a node. To enable the PHY model PHY_UWOP to process events, add the following code to PHY_ProcessEvent (in phy.cpp) to call PhyUWOPTransmissionEnd when messages for PHY_UWOP are received.

Step 9: Modify the generic functions in "QUALNET /libraries/wireless/src/phy.cpp" so that appropriate functions defined in "QUALNET/libraries/user_models/src/phy_UWOP.h" are called when "PHY_UWOP" is chosen as the PHY model running at the interface.

Like the above code, modify the following functions:

1. PHY_StartTransmittingSignal: StartTransmittingSignal performs the following tasks:

- StartTransmittingSignal calls function PHY_GetTransmissionChannel to get the index of the channel on which to transmit the signal. PHY_GetTransmissionChannel is defined in phy.h.
- If PHY is currently receiving a signal, i.e., the status of PHY is PHY_RECEIVING, the PHY model updates the interference power, and resets the receive parameters by calling PhyUWOPUnlockSignal.
- StartTransmittingSignal changes the status of PHY to PHY_TRANSMITTING.
- StartTransmittingSignal calculates the transmission duration of the packet by calling PhyUWOPGetFrameDuration, and adds a Physical Layer header to the packet by calling MESSAGE_AddHeader.
- StartTransmittingSignal calls function PHY_StopListeningToChannel to stop receiving on the channel.
- StartTransmittingSignal calls the communication medium function PROP_ReleaseSignal to transmit the packet. PROP_ReleaseSignal is defined in QUALNET/include/propagation.h.
- StartTransmittingSignal schedules a self-timer of type MSG_PHY_TransmissionEnd to indicate the end of transmission of the packet.

2. PHY_SignalArrivalFromChannel: PhyUWOPSignalArrivalFromChannel performs the following tasks:

- If the PHY model status is PHY_RECEIVING, i.e., the node is already receiving another signal, PhyUWOPSignalArrivalFromChannel calculates the receive power and determines if there are any errors in the portion of the packet received so far by calling function PhyUWOPCheckRxPacketError. PhyUWOPSignalArrivalFromChannel then adds the receive power to the interference power.
- If the PHY model status is PHY_IDLE or PHY_SENSING, PhyUWOPSignalArrivalFromChannel calculates the interference power and received power.
- If the received power is greater than the receiver sensitivity, PhyUWOPSignalArrivalFromChannel locks on to the signal by calling PhyUWOPLockSignal, changes status to PHY_RECEIVING, and informs the MAC Layer of the status change by calling PhyUWOPReportExtendedStatusToMac.
- If the received power is less than the receiver sensitivity, PhyUWOPSignalArrivalFromChannel calls function PhyUWOPCarrierSensing to determine if the signal strength is high enough to trigger a status change. If a status change is triggered, PhyUWOPSignalArrivalFromChannel updates the status and informs the MAC Layer of the status change by calling PhyUWOPReportStatusToMac.

3. PHY_SignalEndFromChannel : PhyUWOPSignalEndFromChannel defined in phy.cpp performs the following tasks:

- PhyUWOPSignalEndFromChannel checks if there are any errors in the received packet by calling PhyUWOPCheckRxPacketError.
- If the PHY model status is PHY_RECEIVING and the received signal is the one that the PHY model had locked on to, PhyUWOPSignalEndFromChannel stops receiving the signal and calls PhyUWOPUnlockSignal.
- PhyUWOPSignalEndFromChannel calls PhyUWOPCarrierSensing and changes the PHY model status to PHY_SENSING or PHY_IDLE depending on the interference power.

- If the packet was received without any errors, PhyUWOPSignalEndFromChannel removes the Physical Layer header and sends the packet to the MAC Layer by calling MAC_ReceivePacketFromPhy. MAC_ReceivePacketFromPhy is implemented in QUALNET/main/mac.cpp.
- If the packet was received with errors, PhyUWOPSignalEndFromChannel reports the status change to the MAC Layer by calling PhyUWOPReportStatusToMac and drops the packet.
- If the PHY model status is not PHY_RECEIVING or the received signal is not the one that the PHY model had locked on to, PhyUWOPSignalEndFromChannel updates the interference power. If the PHY model status is not PHY_RECEIVING, PhyUWOPSignalEndFromChannel calls PhyUWOPCarrierSensing and changes the PHY model status to PHY_SENSING or PHY_IDLE depending on the interference power.
- If the PHY model status changes, PhyUWOPSignalEndFromChannel reports the status change to the MAC Layer by calling PhyUWOPReportStatusToMac.

4. PHY_GetTxDataRate
5. PHY_GetRxDataRate
6. PHY_GetTransmissionDuration
7. PHY_SetTransmitPower
8. PHY_GetTransmitPower

Step 10: Modify the function “PHY_Finalize” in the source file “qualnet/libraries/user_models/src/phy.cpp” as follows for the finalization function “PhyUWOPFinalize” to be called at the end of simulation.

Step 11: Using the IEEE 802.11a finalization function template, write PhyUWOPFinalize(). PhyUWOPFinalize() is implemented in phy_UWOP.cpp.

Step 12: The file qualnet /libraries/wireless/src/prop_range.cpp implements the radio-range program, which calculates the likely propagation range of a node, under no interference conditions, using the parameters specified in the configuration file. Modify this file to incorporate PHY_UWOP.

Step 13: The final step in integrating your PHY model into QualNet is to add the source file to the QualNet source tree and compile. Before recompiling delete all the object files.

Step 14: Finally, Modify the PHY description file, "phy_layer.prt" in the directory "qualnet/gui/settings/protocol_models/", to integrate the EM radio PHY model into the QualNet GUI.

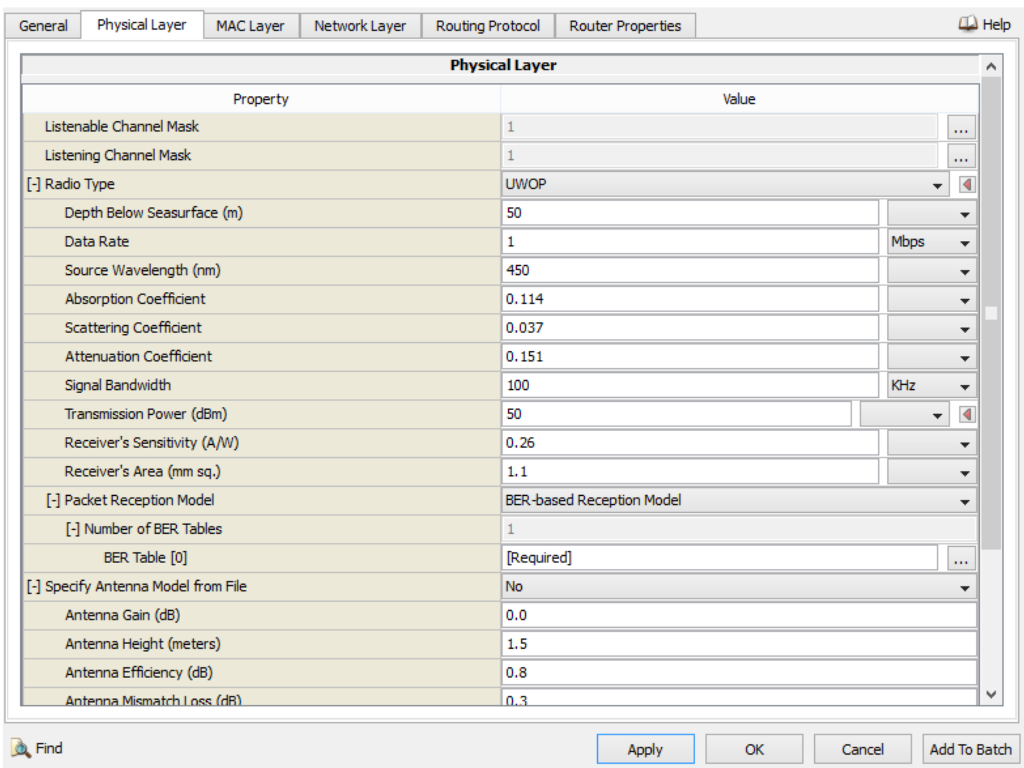

Fig 4.7 Screenshot of Optical Physical Layer Model in Qualnet

4.7 *Physical layer modelling for Acoustic communication*

Step 1: Create header and source files namely phy_UWAP.h and phy_UWAP.cpp. Place them in the folder QUALNET/libraries/user_models/src.

Step 2: The header file, phy_UWAP.h, should contain the following:

- Constant definitions
- Data structure definitions
- Prototypes for interface functions in the source file, phy_UWAP.cpp

Step 3: Specify the Physical model created, PHY_UWAP, in the enumeration in QUALNET/include/phy.h

Step 4: Modify the function “AddNodeToSubnet” in “qualnet/main/mac.cpp” as follows so that the PHY initialization function “PHY_CreateAPhyForMac” is called and “phyModel” is set to “PHY_UWAP” when “PHY_UWAP” is specified as the PHY model.

Step 5: Modify the function “PHY_CreateAPhyForMac” in “qualnet/libraries/wireless/ src/phy.cpp” as follows so that when the function is called, the channel and PHY parameters specified in the configuration file are read and stored, the used reception model (“SNR_BASED” in this work) and receiver parameters are set, and the initialization function (i.e., “PhyUWAPInit”) is executed.

Step 6. Modify the PHY event dispatch function “PHY_ProcessEvent” in source file “qualnet/libraries/wireless/src/phy.cpp” as follows to trigger a timer event indicating the end of transmission of a packet in a node (i.e.,“MSG_PHY_TransmissionEnd”) (handled by the function “PhyUWAPTransmissionEnd” in “qualnet/addons/UWEMMAC/phy_UWAP.h”).

Step 7. Modify the generic functions in “qualnet/libraries/wireless/src/phy.cpp” so that appropriate functions defined in “qualnet/addons/UWEMMAC/phy_UWAP.h” are called when “PHY_UWAP” is chosen as the PHY model running at the interface.

Step 8. Modify the GUI accordingly to see the desired changes and take the user inputs.

4.8 Network architecture implemented in Qualnet

The network of 100 nodes is constructed in Qualnet 5.0 to study the effect of underwater environment in optical and acoustic sensor network communication. The sensor network architectured using the GUI of the Qualnet is shown in Fig.4.8 and Fig.4.9 and its properties are specified in Table 4.1.

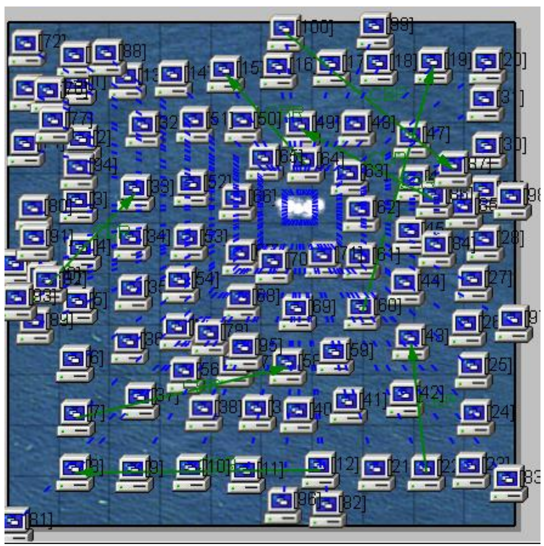

Fig 4.8 Sensor Network Architecture with 100 nodes in 2D

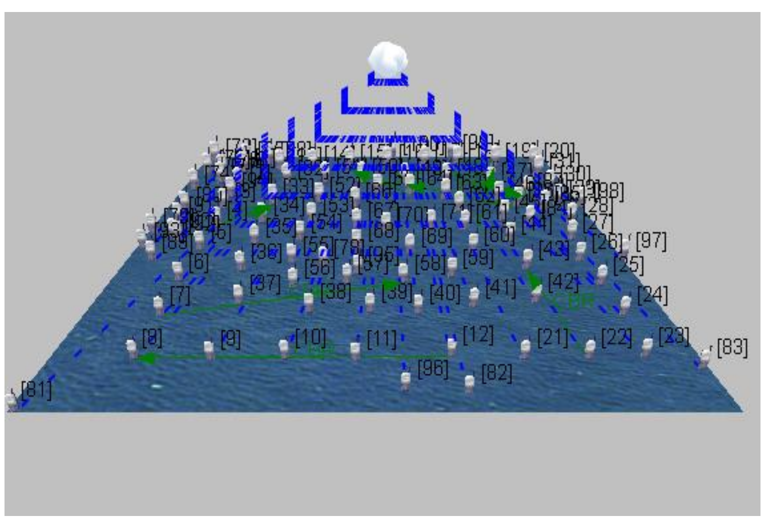

Fig 4.9 Sensor Network Architecture with 100 nodes in 3D

The complete network is placed onto a cartesian coordinate plane i.e 200×200 has altitude ranging from 0 m above sea level to 100 m below sea level. One channel is considered for communication and its frequency is selected according to the type of communication i.e optical or acoustic.

The modelling of communication medium is done by specifying Pathloss, Shadowing and Fading Model. For our work, we have chosen Fading model as 'Ricean model' and the shadowing model as 'None'. The Pathloss model is customized and designed for both the optical and acoustic networks. They are named as 'Underwater Optical Model' and 'Underwater Acoustic Model'. Mobility model is considered as the nodes in the water body are subjected to movement due to the turbulence in the environment. Therefore, we take the mobility model as the Random Waypoint model. We have chosen the position granularity i.e the distance by which node moves 1mm in a single step for a simulation of 5 days. The network works on different protocols for different layers. While Network layer uses IPv4 and MAC layer uses CSMA, the Routing protocol chosen is AODV (Ad-Hoc On Demand Distance Vector). All the specifications of the network are mentioned in Table 4.1.

Table 4.1 Network Specifications

1.	**Scenario Properties–> General**	**Value**
1.a	Simulation Time	5 days
1.b	Real Time	45 minutes
1.c	Background Image	GUI image of water
2.	**Scenario Properties–> Channel Properties**	
2.a	Coordinate System	Cartesian X=200m Y=200m
2.b	Altitude Range (in m)	Above Sea level=0 Below Sea level=100
2.c	Weather Mobility Interval	100 msec
2.d	Number of Channels	1

2.e	Channel Frequency	560 THz (For green light) 40 KHz (for acoustic communication)
2.f	Pathloss Model	Underwater Optical Model/Underwater Acoustic Model
2.g	Shadowing Model	None
2.h	Fading Model	Ricean
2.i	Propagation limit	See Table 4.2
2.j	Maximum Propagation Distance (Maximum Distance for which a node's transmission is considered for communication. If this parameter= 0.0, then it is not considered in the estimation of a node's propagation range i.e maximum distance is effectively infinity)	See Table 4.3
2.k	Propagation Communication Proximity (This parameter should be set to the approximate optical/acoustic range)	See Table 4.3
3.	**Node Properties–> Mobility and Placement**	
3.a	Mobility Waypoint	Random Waypoint
3.b	Pause Time	2 min
3.c	Minimum Speed	0 m/sec
3.d	Maximum Speed	0.1 m/sec
3.e	Position Granularity (Distance by which node moves in a single step)	0.001 m
4.	**Network Layer Protocol**	**IPv4**
5.	**Routing Protocol**	**AODV**
5.a	Network Diameter (Maximum possible number of hops between 2 nodes in the	5

	N/W)	
5.b	Node Traversal Time (It specifies the estimate of average 1-hop traversal time for packets and should include queuing transmission, propagation and other delays.)	40 msec
5.c	Active Route Timeout (Expiry timeout of an active route)	3 sec
6.	**Subnet Properties–> Physical Layer**	
6.a	Radio Type	UWOP/UWAP
6.b	Data Rate	Case 1: 10 Mbps Case 2: 2 Mbps Case 3: 100 Kbps Case 4: 9.2 Kbps
6.c	Temperature	305 K
6.d	Noise Factor	0 dB
6.e	Energy Model	Mica-Motes

There are different values which need to be set according to the type of link desired between nodes. Propagation limit is the threshold for delivering the signal to the nodes. Signal below this limit (of power) are not delivered to the nodes. This parameter is meant for optimizing simulation performance. Table 4.2 shows the calculated values of Propagation Limit for both optical and acoustic communication for our scenario.

Table 4.2 Propagation Limit Specification

Type of Communication	Propagation Limit (dBm)
Optical	$I_{t=}I_0 e^{-c(\lambda)z}$, with λ=490 nm (for pure sea water), z=40m (depth in water), I_0=0.1W (for LED source), We get I_t=0.045W, in dBm=16.53 dBm
Acoustic	A(d, f) = $A_0\ d^k a(f)^d$, where A_0 is a unit-normalizing constant that includes fixed losses, $a(f)$

	is the absorption coefficient and k is the spreading factor. In the case of practical spreading, k = 1.5. For f=40 KHz, $a(f)$=15 dB/km from the thorp's formula. For d=200m we get A(d,f)=0.153. For Transmitted power=15W, therefore power received= 15×(1-0.153)=12.7W= 41.038 dBm

Further, other parameters like Maximum Propagation Distance and Propagation Communication Proximity are also required to determine the communication links. Maximum Propagation Distance is the maximum distance for which a node's transmission is considered for communication. Similarly, Here, we have set the Propagation Communication Proximity according to the approximate range given by Jingjing Wang et al. 2017 in their work. The values of both the parameters for different transmission rates are shown in Table 4.3.

Table 4.3 Propagation Communication Proximity for corresponding Data Rate value

Transmission Rate	**Maximum Propagation Distance**	**Considered Propagation Communication Proximity**
10 Mbps	Infinite	30 m
2 Mbps	Infinite	50 m
100 Kbps	Infinite	80 m
9.2 Kbps	Infinite	200 m

All these specifications are used to depict the channel which acts as a communication medium. The efficiency of the communication depends on the channel therefore it is important to model near-to-real channel. All the performance metric results corresponding to the specifications mentioned in Table 4.1, 4.2 and 4.3 are discussed in chapter-5.