Diagram Documentation

Primary Use Cases - *(Use Case diagram)*

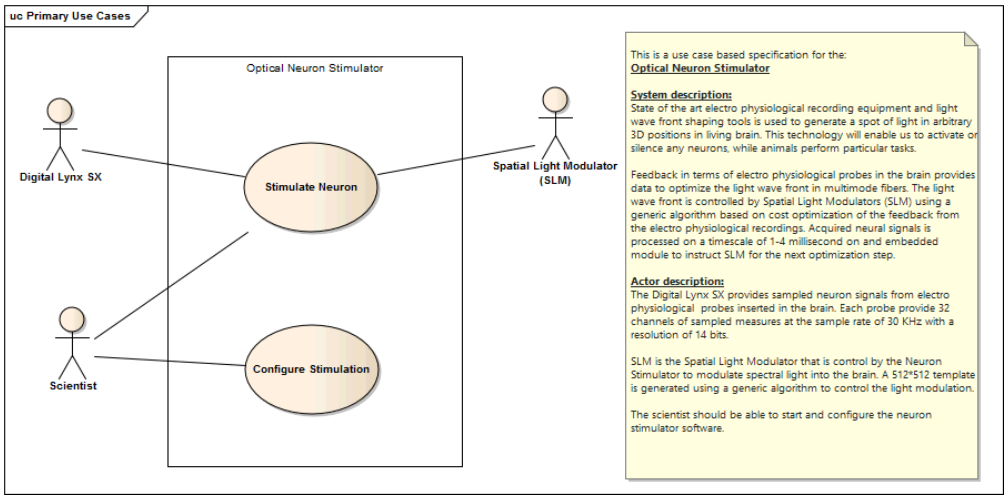

Figure: 1

Logical View - *(Class diagram)*

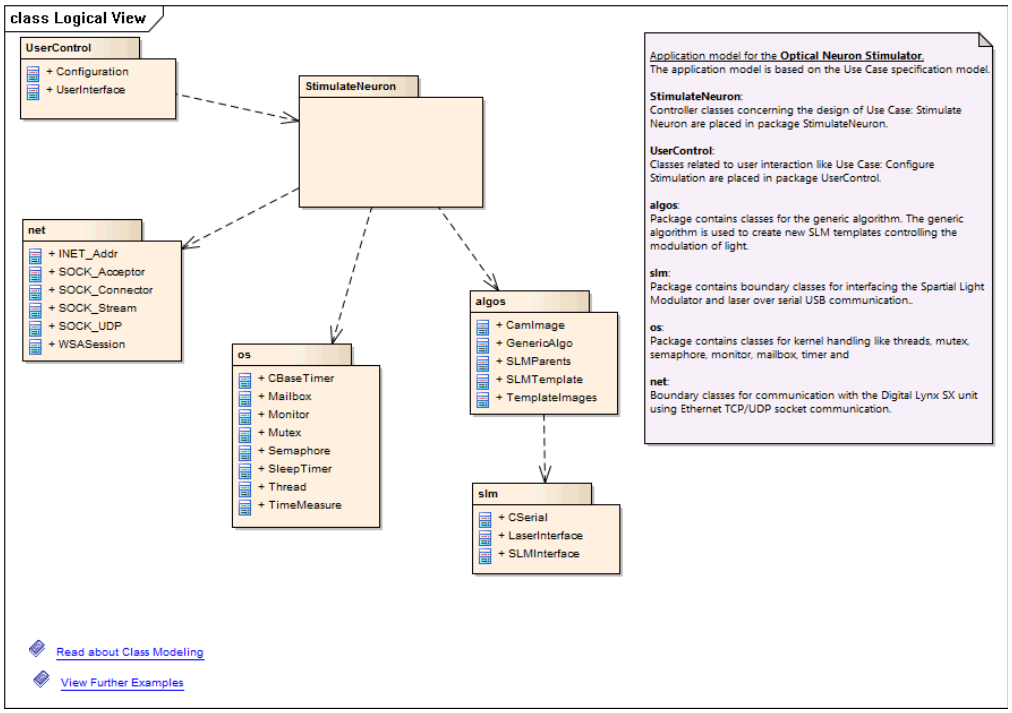

Figure: 2

StimulateNeuron Classes - *(Class diagram)*

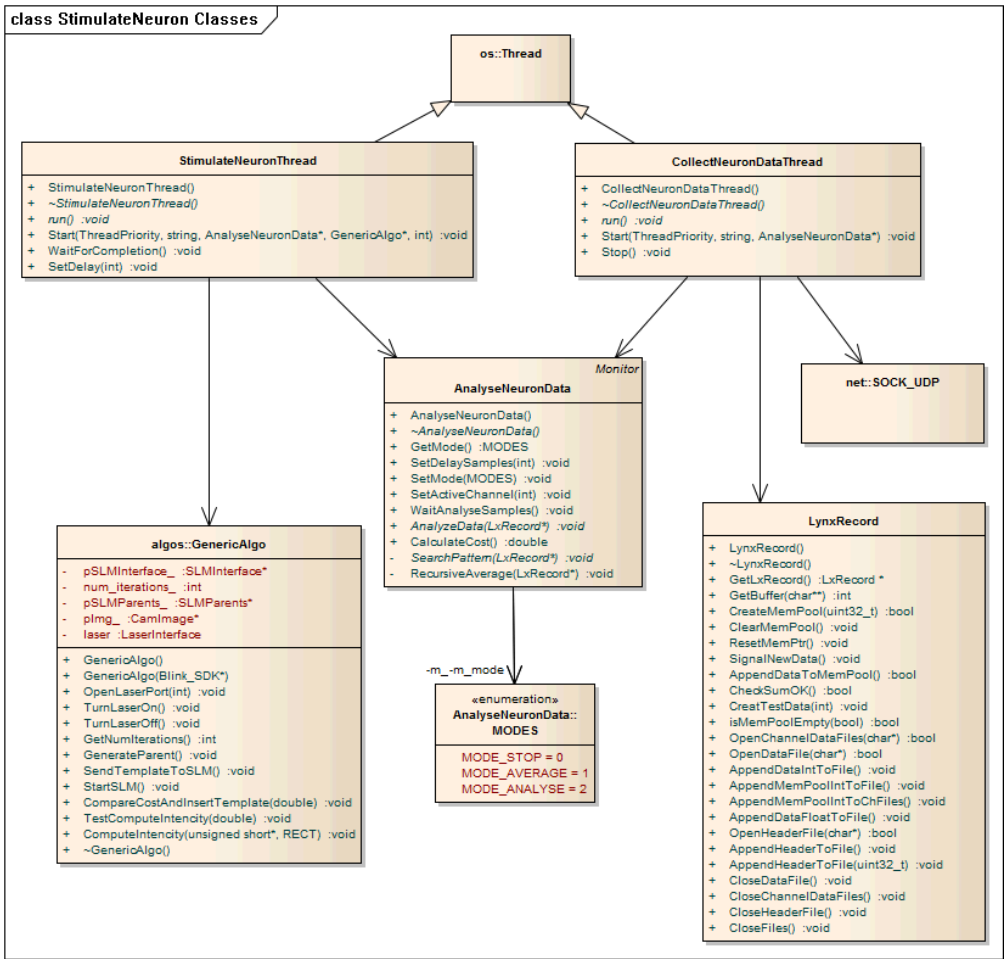

Figure: 3

Generate Templates and Control SLM - *(Interaction diagram)*

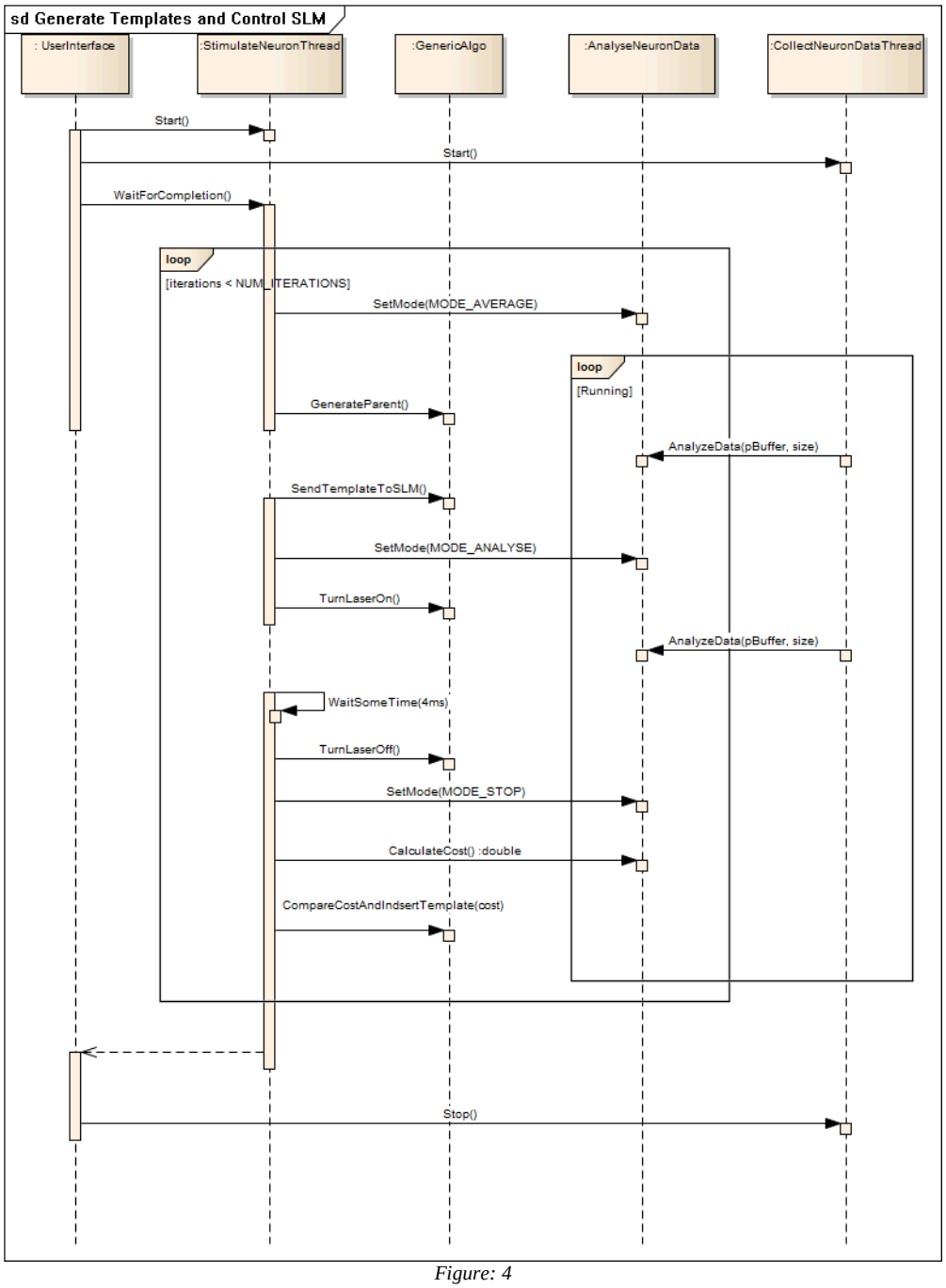
sd Generate Templates and Control SLM
: UserInterface
:StimulateNeuronThread
:GenericAlgo
:AnalyseNeuronData
:CollectNeuronDataThread
Start()
Start()
WaitForCompletion()
loop
[iterations < NUM_ITERATIONS]
SetMode(MODE_AVERAGE)
loop
[Running]
GenerateParent()
AnalyzeData(pBuffer, size)
SendTemplateToSLM()
SetMode(MODE_ANALYSE)
TurnLaserOn()
AnalyzeData(pBuffer, size)
WaitSomeTime(4ms)
TurnLaserOff()
SetMode(MODE_STOP)
CalculateCost() :double
CompareCostAndIndsertTemplate(cost)
Stop()

Figure: 4

Collect and Analyze Neuron Data - *(Interaction diagram)*

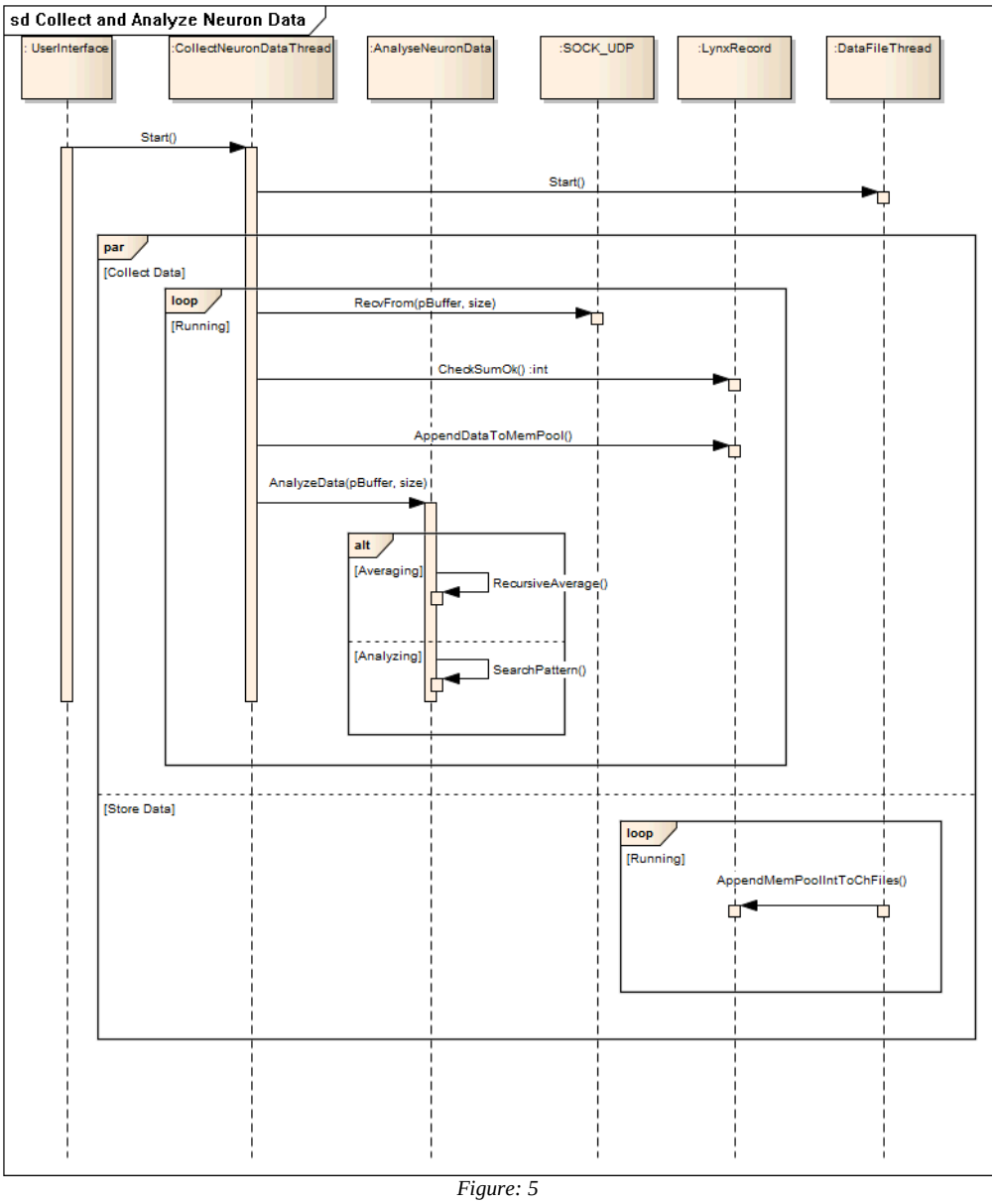

Figure: 5

Neuron Data File Thread - *(Class diagram)*

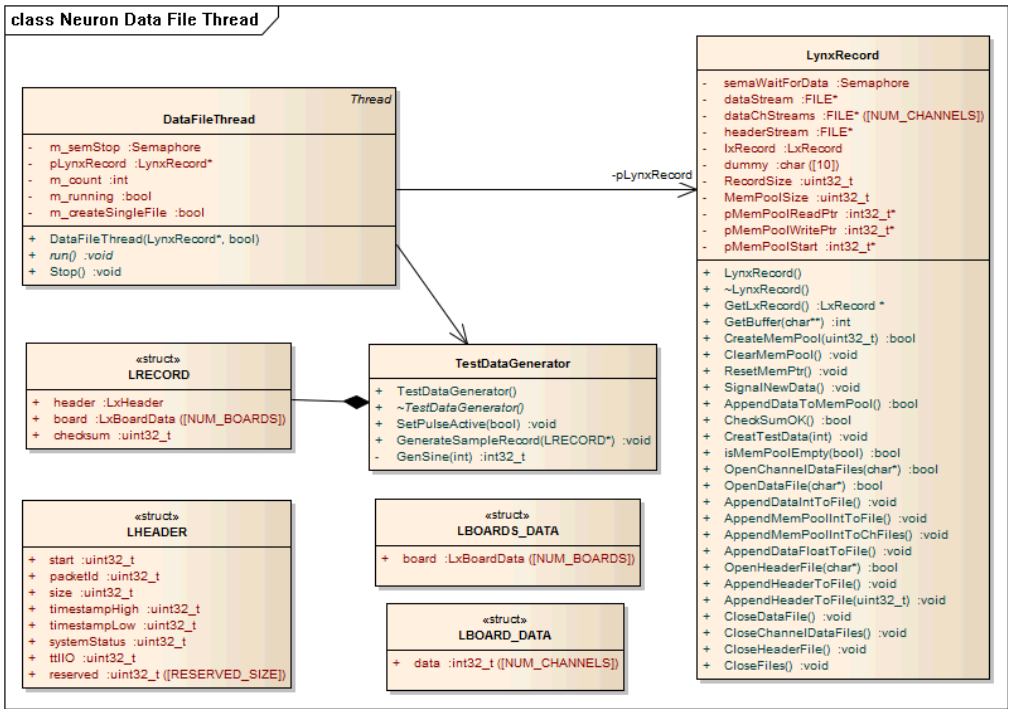

Figure: 6

Stimulate Neuron States - *(StateMachine diagram)*

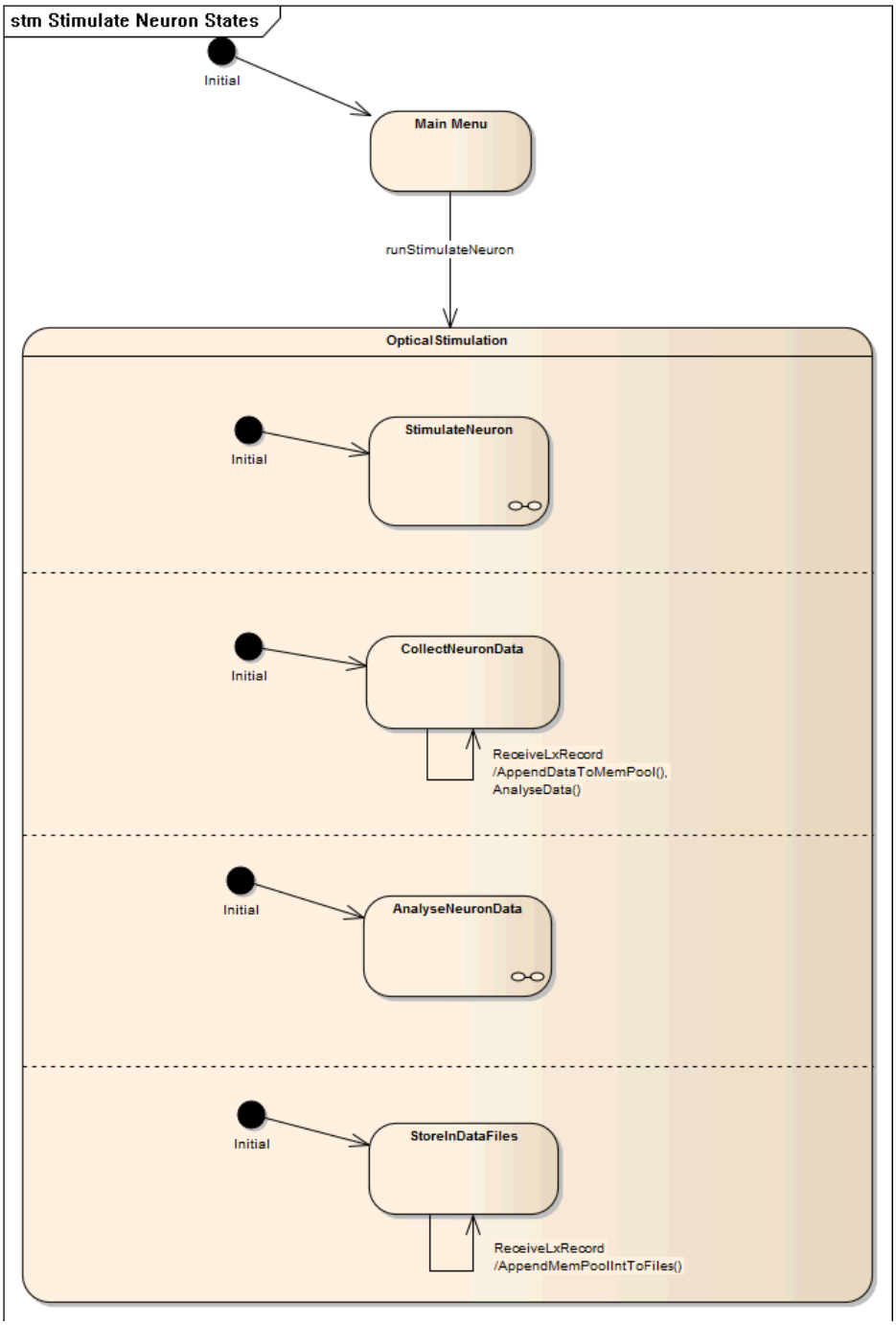
stm Stimulate Neuron States
Initial
Main Menu
runStimulateNeuron
Optical Stimulation
Initial
StimulateNeuron
Initial
CollectNeuronData
ReceiveLxRecord
/AppendDataToMemPool(),
AnalyseData()
Initial
AnalyseNeuronData
Initial
StoreInDataFiles
ReceiveLxRecord
/AppendMemPoolIntToFiles()

Figure: 7

AnalyseNeuronData - *(StateMachine diagram)*

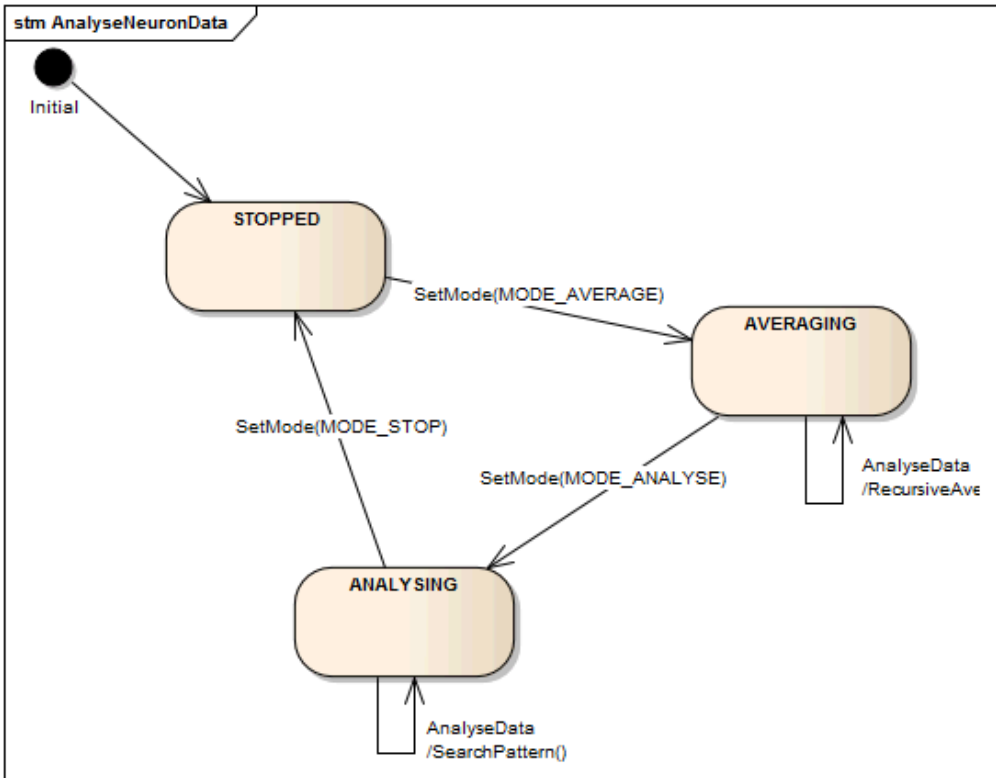

Figure: 8

StimulateNeuron - *(StateMachine diagram)*

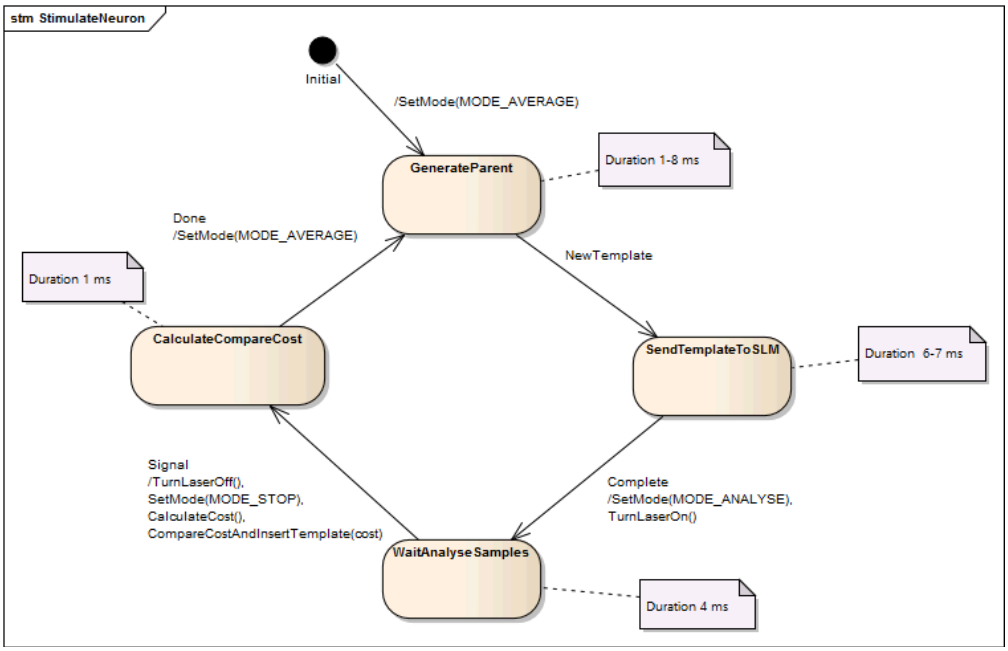

Figure: 9

UserControl - *(Class diagram)*

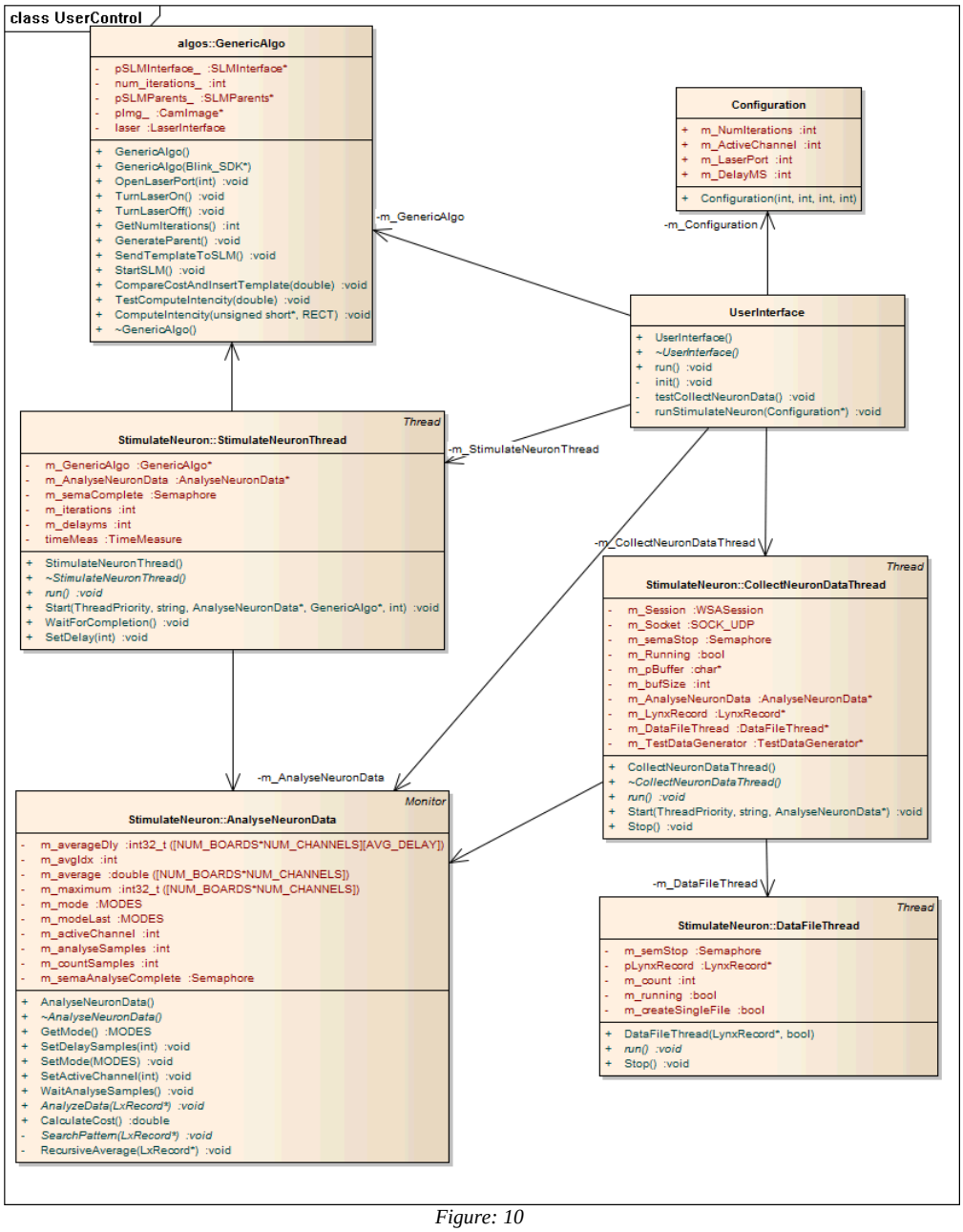
class UserControl
algos::GenericAlgo
- pSLMInterface_ :SLMInterface*
- num_iterations_ :int
- pSLMParents_ :SLMParents*
- pImg_ :CamImage*
- laser :LaserInterface
+ GenericAlgo()
+ GenericAlgo(Blink_SDK*)
+ OpenLaserPort(int) :void
+ TurnLaserOn() :void
+ TurnLaserOff() :void
+ GetNumIterations() :int
+ GenerateParent() :void
+ SendTemplateToSLM() :void
+ StartSLM() :void
+ CompareCostAndInsertTemplate(double) :void
+ TestComputeIntencity(double) :void
+ ComputeIntencity(unsigned short*, RECT) :void
+ ~GenericAlgo()
Configuration
+ m_NumIterations :int
+ m_ActiveChannel :int
+ m_LaserPort :int
+ m_DelayMS :int
+ Configuration(int, int, int, int)
-m_GenericAlgo
-m_Configuration
UserInterface
+ UserInterface()
+ ~UserInterface()
+ run() :void
- init() :void
- testCollectNeuronData() :void
- runStimulateNeuron(Configuration*) :void
Thread
StimulateNeuron::StimulateNeuronThread
- m_GenericAlgo :GenericAlgo*
- m_AnalyseNeuronData :AnalyseNeuronData*
- m_semaComplete :Semaphore
- m_iterations :int
- m_delayms :int
- timeMeas :TimeMeasure
+ StimulateNeuronThread()
+ ~StimulateNeuronThread()
+ run() :void
+ Start(ThreadPriority, string, AnalyseNeuronData*, GenericAlgo*, int) :void
+ WaitForCompletion() :void
+ SetDelay(int) :void
-m_StimulateNeuronThread
-m_CollectNeuronDataThread
Thread
StimulateNeuron::CollectNeuronDataThread
- m_Session :WSASession
- m_Socket :SOCK_UDP
- m_semaStop :Semaphore
- m_Running :bool
- m_pBuffer :char*
- m_bufSize :int
- m_AnalyseNeuronData :AnalyseNeuronData*
- m_LynxRecord :LynxRecord*
- m_DataFileThread :DataFileThread*
- m_TestDataGenerator :TestDataGenerator*
+ CollectNeuronDataThread()
+ ~CollectNeuronDataThread()
+ run() :void
+ Start(ThreadPriority, string, AnalyseNeuronData*) :void
+ Stop() :void
-m_AnalyseNeuronData
Monitor
StimulateNeuron::AnalyseNeuronData
- m_averageDly :int32_t ([NUM_BOARDS*NUM_CHANNELS][AVG_DELAY])
- m_avgIdx :int
- m_average :double ([NUM_BOARDS*NUM_CHANNELS])
- m_maximum :int32_t ([NUM_BOARDS*NUM_CHANNELS])
- m_mode :MODES
- m_modeLast :MODES
- m_activeChannel :int
- m_analyseSamples :int
- m_countSamples :int
- m_semaAnalyseComplete :Semaphore
+ AnalyseNeuronData()
+ ~AnalyseNeuronData()
+ GetMode() :MODES
+ SetDelaySamples(int) :void
+ SetMode(MODES) :void
+ SetActiveChannel(int) :void
+ WaitAnalyseSamples() :void
+ AnalyzeData(LxRecord*) :void
+ CalculateCost() :double
- SearchPattern(LxRecord*) :void
- RecursiveAverage(LxRecord*) :void
-m_DataFileThread
Thread
StimulateNeuron::DataFileThread
- m_semStop :Semaphore
- pLynxRecord :LynxRecord*
- m_count :int
- m_running :bool
- m_createSingleFile :bool
+ DataFileThread(LynxRecord*, bool)
+ run() :void
+ Stop() :void

Figure: 10

Generic Algorithm Classes - *(Class diagram)*

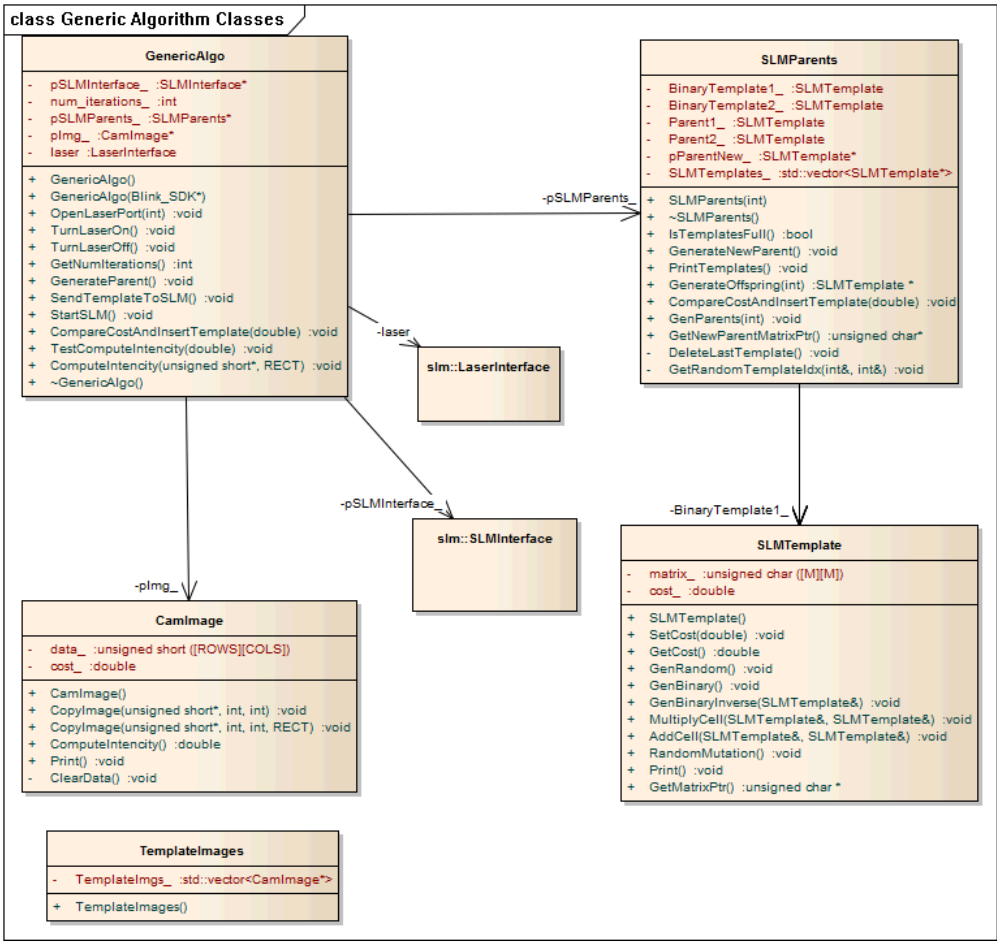

Figure: 11

SLM and Laser Interface Classes - *(Class diagram)*

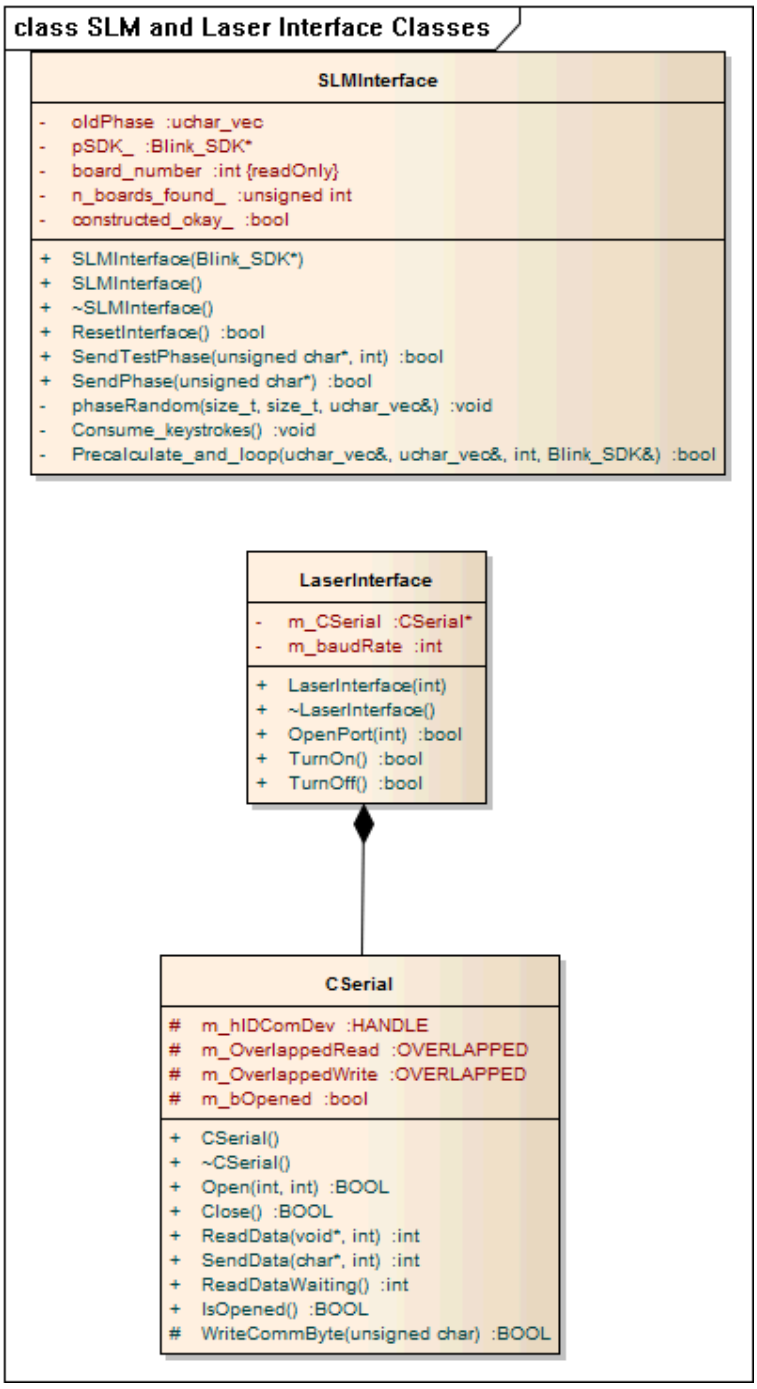
class SLM and Laser Interface Classes
SLMInterface
- oldPhase :uchar_vec
- pSDK_ :Blink_SDK*
- board_number :int {readOnly}
- n_boards_found_ :unsigned int
- constructed_okay_ :bool
+ SLMInterface(Blink_SDK*)
+ SLMInterface()
+ ~SLMInterface()
+ ResetInterface() :bool
+ SendTestPhase(unsigned char*, int) :bool
+ SendPhase(unsigned char*) :bool
- phaseRandom(size_t, size_t, uchar_vec&) :void
- Consume_keystrokes() :void
- Precalculate_and_loop(uchar_vec&, uchar_vec&, int, Blink_SDK&) :bool
LaserInterface
- m_CSerial :CSerial*
- m_baudRate :int
+ LaserInterface(int)
+ ~LaserInterface()
+ OpenPort(int) :bool
+ TurnOn() :bool
+ TurnOff() :bool
CSerial
m_hIDComDev :HANDLE
m_OverlappedRead :OVERLAPPED
m_OverlappedWrite :OVERLAPPED
m_bOpened :bool
+ CSerial()
+ ~CSerial()
+ Open(int, int) :BOOL
+ Close() :BOOL
+ ReadData(void*, int) :int
+ SendData(char*, int) :int
+ ReadDataWaiting() :int
+ IsOpened() :BOOL
# WriteCommByte(unsigned char) :BOOL

Figure: 12

Kernel Classes - *(Class diagram)*

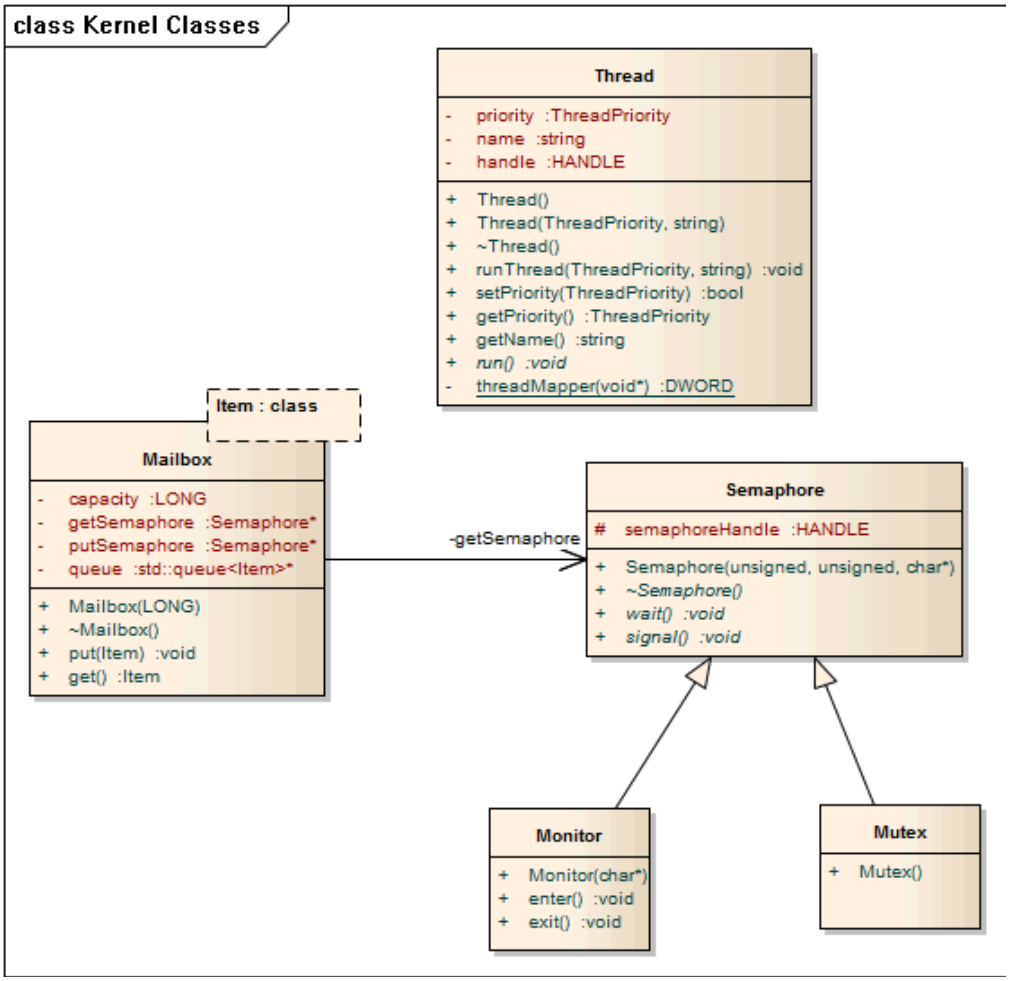

Figure: 13

Timer Classes - *(Class diagram)*

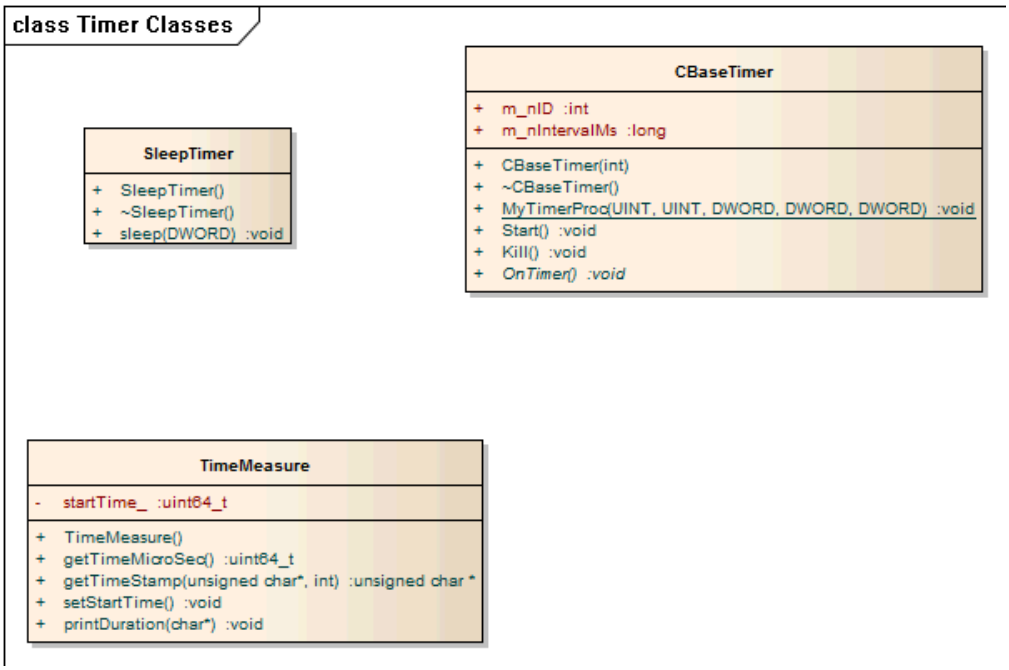

Figure: 14

Network Classes - *(Class diagram)*

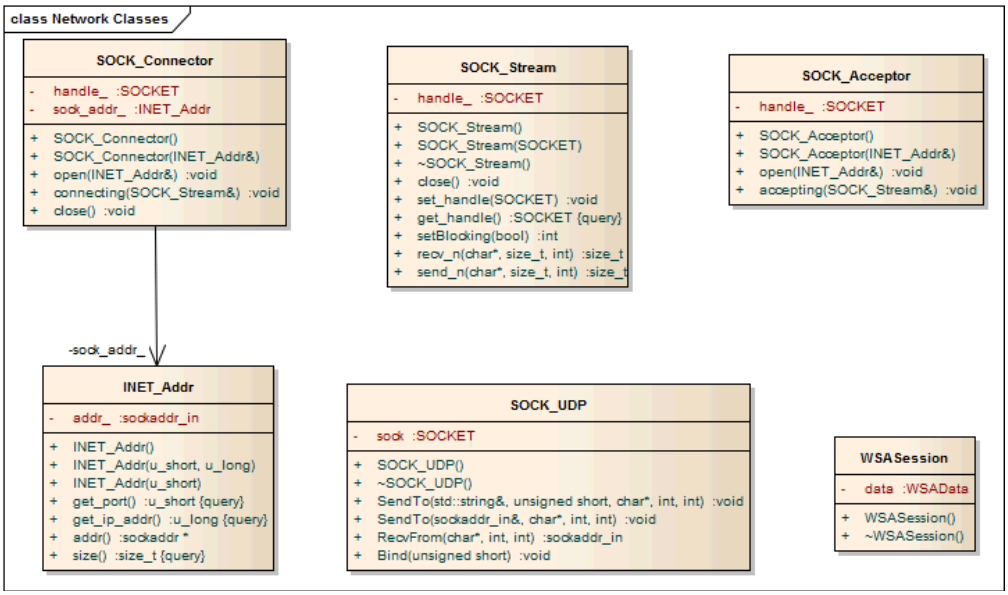
class Network Classes
SOCK_Connector
- handle_ :SOCKET
- sock_addr_ :INET_Addr
+ SOCK_Connector()
+ SOCK_Connector(INET_Addr&)
+ open(INET_Addr&) :void
+ connecting(SOCK_Stream&) :void
+ close() :void
-sock_addr_
SOCK_Stream
- handle_ :SOCKET
+ SOCK_Stream()
+ SOCK_Stream(SOCKET)
+ ~SOCK_Stream()
+ close() :void
+ set_handle(SOCKET) :void
+ get_handle() :SOCKET {query}
+ setBlocking(bool) :int
+ recv_n(char*, size_t, int) :size_t
+ send_n(char*, size_t, int) :size_t
SOCK_Acceptor
- handle_ :SOCKET
+ SOCK_Acceptor()
+ SOCK_Acceptor(INET_Addr&)
+ open(INET_Addr&) :void
+ accepting(SOCK_Stream&) :void
INET_Addr
- addr_ :sockaddr_in
+ INET_Addr()
+ INET_Addr(u_short, u_long)
+ INET_Addr(u_short)
+ get_port() :u_short {query}
+ get_ip_addr() :u_long {query}
+ addr() :sockaddr *
+ size() :size_t {query}
SOCK_UDP
- sock :SOCKET
+ SOCK_UDP()
+ ~SOCK_UDP()
+ SendTo(std::string&, unsigned short, char*, int, int) :void
+ SendTo(sockaddr_in&, char*, int, int) :void
+ RecvFrom(char*, int, int) :sockaddr_in
+ Bind(unsigned short) :void
WSASession
- data :WSAData
+ WSASession()
+ ~WSASession()

Figure: 15