Smart Sensor Network: Services Functions

IEEE Standard P21451-1

Faculty Advisor:
Dr. John L. Schmalzel

Graduate Assistants:
Tom Morris **Russell Trafford**

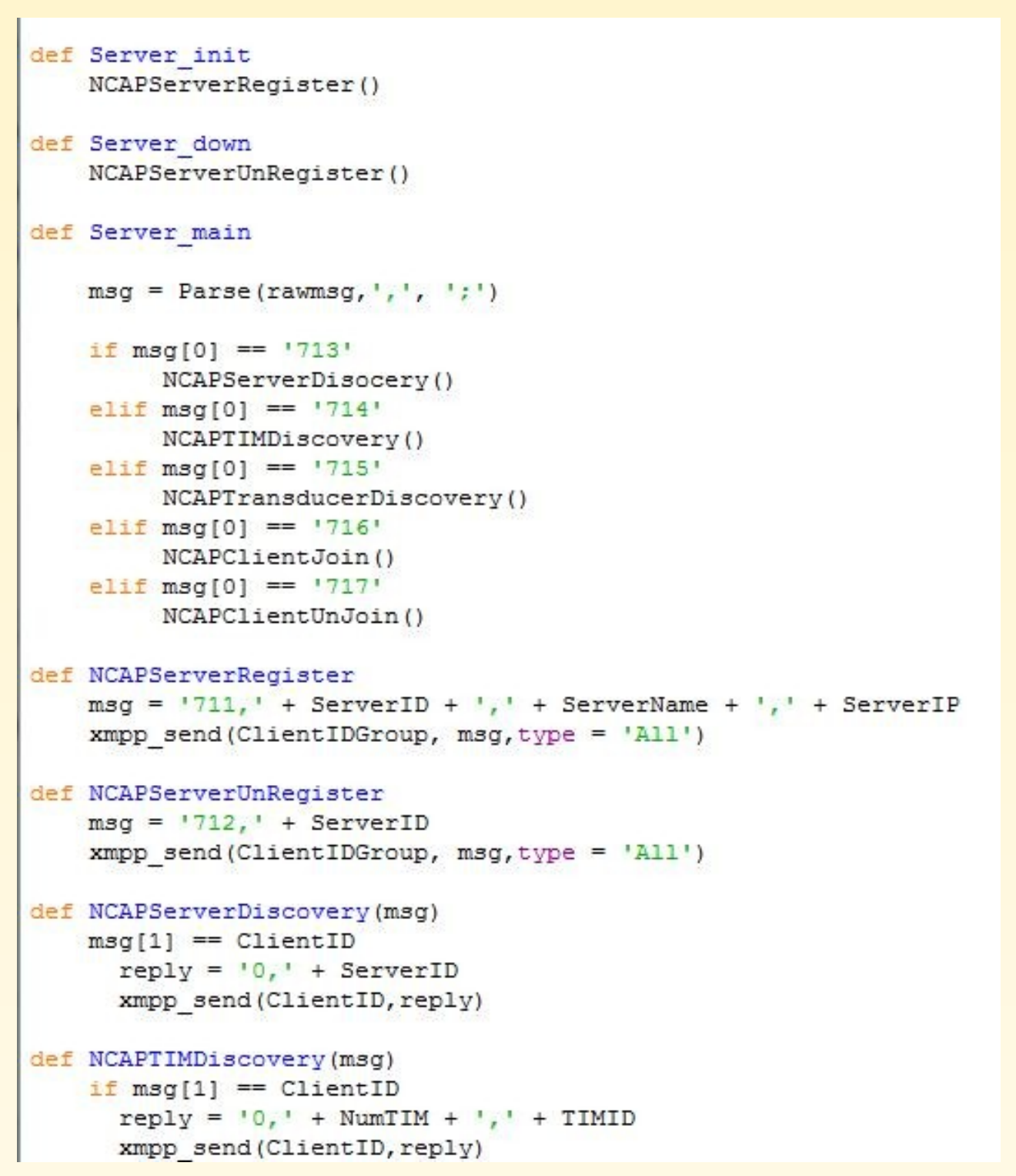
Research Assistants:
Brian Finch **Eric Guidarelli**
Keith Hall **Jacob Harris**
Anas Muhamed **Matt Oldland**
Nick Parisi **Tom Stoudt**

Abstract

The P21451-1 Standard defines a set of common network communication protocols for IEEE 1451 smart transducers and has five main services that are illustrated in the table below: Identification, Transducers Access, TEDs Access Services, Event Notification, and Transducer Management Services. The network also identifies protocols and performs other services. The basis of the standard defines the communication between clients, servers, and transducer interface modules (TIMs). The primary communication method used in this project were XMPP (Extensible Messaging Presence Protocol) for communication between the NCAP Client and NCAP Server, and UART (Universal Asynchronous Receiver/Transmitter) for communication between the NCAP Server and the TIM. Previously, UDP communication was used for as the basis for all communication between the nodes of the network, however, this communication method was slower and required all the nodes to be connected on the same router of a local network. Using XMPP, the nodes of the network do not have to be connected to the same network as each other as the nodes are connected onto the internet.

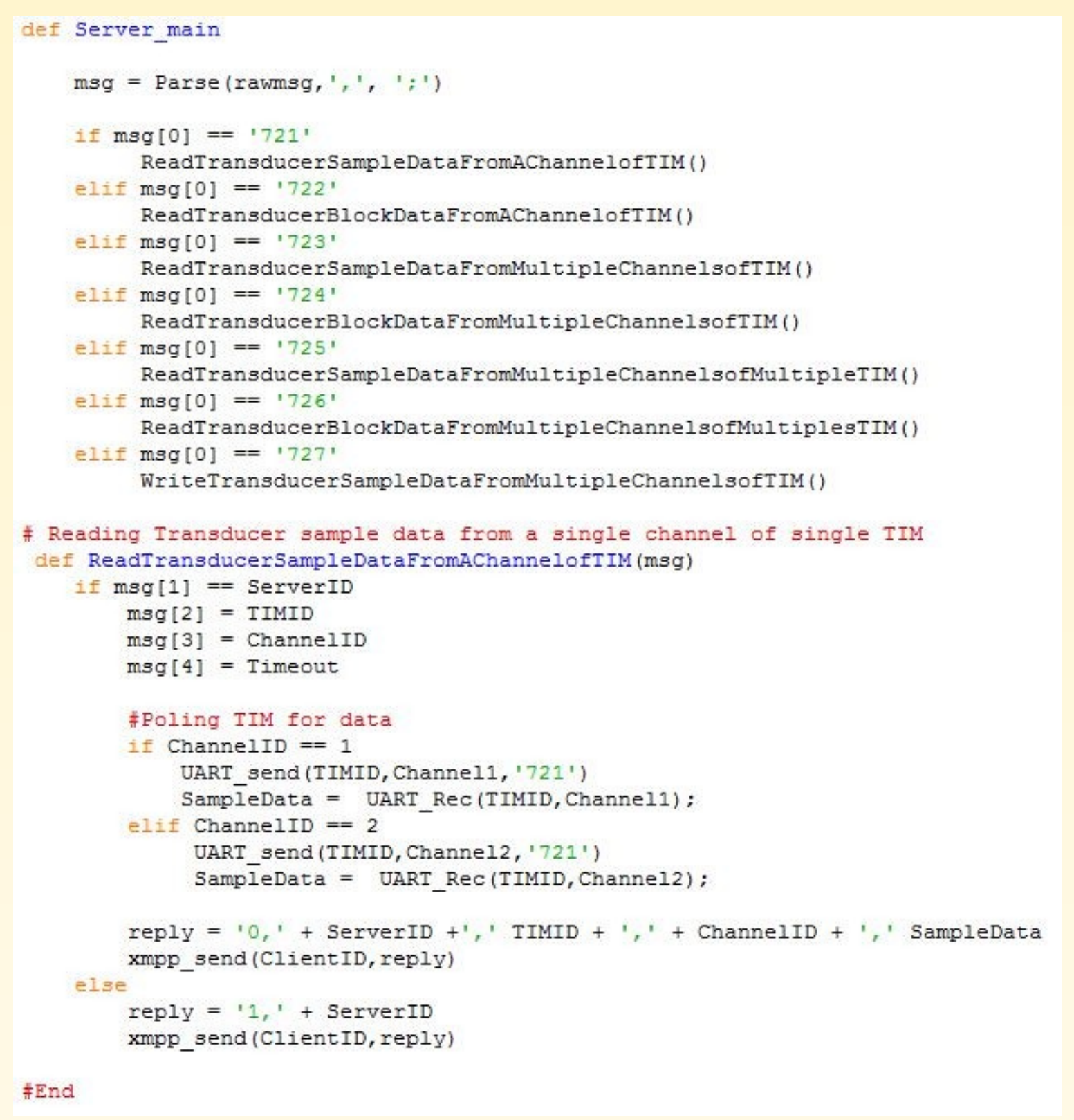
P21451-1 Services	
Identification	The process in which the client, servers, and TIMs discover and establish communication between each other.
Transducer Access	The process in which data is read from the transducer and conveyed to the client from the server.
TEDS Access	The process in which the client/servers read and write from the TEDs of the transducer.
Event Notification	The process in which a client is alerted by a server that a new TIM has been connected/disconnected to the network. The client is then alerted when a sensor alert has occurred.
Transducer Management	The process in which all transducers are configured, evaluated, diagnosed, located, synced, and calibrated.

Identification Services:



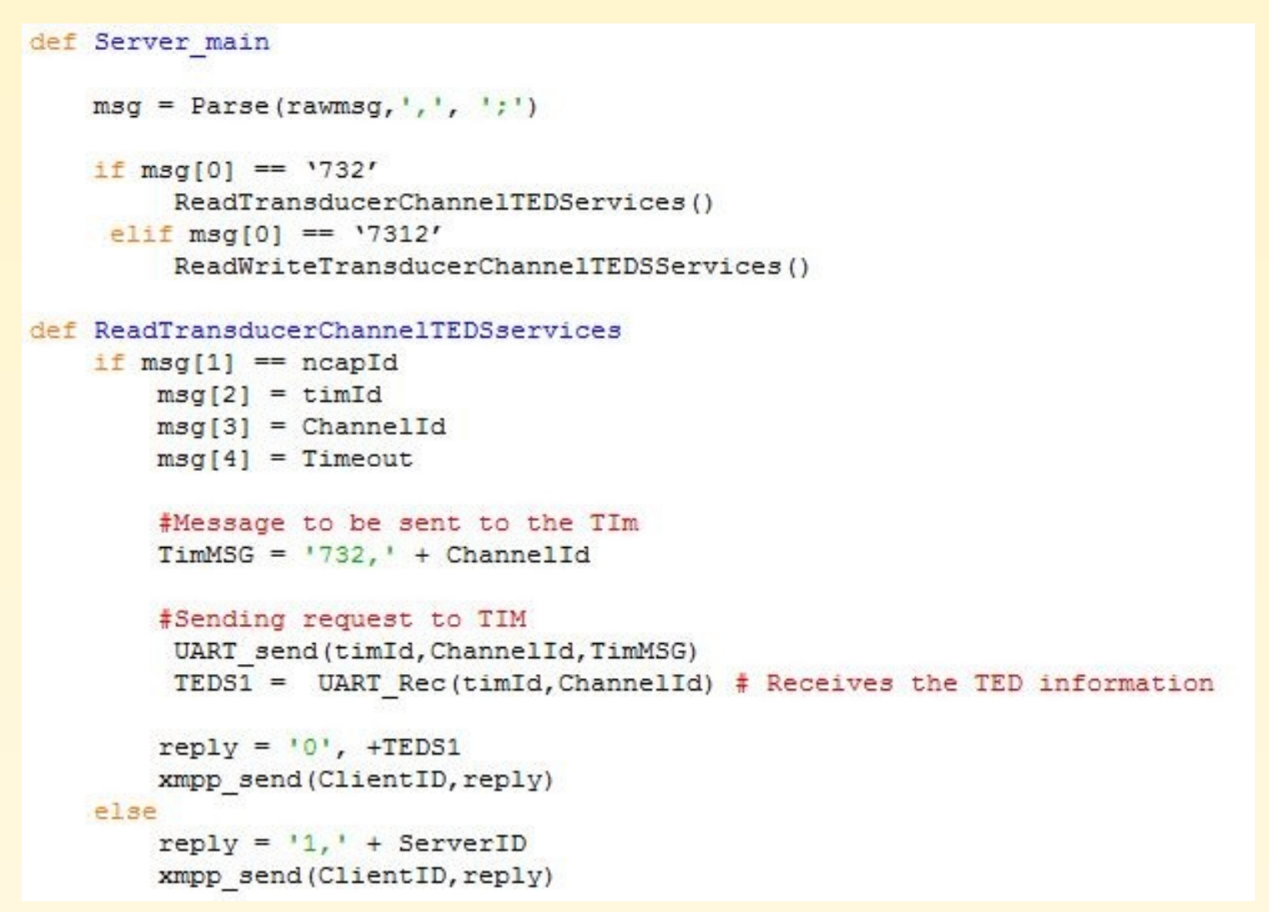
```
def Server_init
    NCAPServerRegister()

def Server_down
    NCAPServerUnRegister()

def Server_main

    msg = Parse(rawmsg,',', ';')

    if msg[0] == '713'
         NCAPServerDisocery()
    elif msg[0] == '714'
         NCAPTIMDiscovery()
    elif msg[0] == '715'
         NCAPTransducerDiscovery()
    elif msg[0] == '716'
         NCAPClientJoin()
    elif msg[0] == '717'
         NCAPClientUnJoin()

def NCAPServerRegister
    msg = '711,' + ServerID + ',' + ServerName + ',' + ServerIP
    xmpp_send(ClientIDGroup, msg,type = 'All')

def NCAPServerUnRegister
    msg = '712,' + ServerID
    xmpp_send(ClientIDGroup, msg,type = 'All')

def NCAPServerDiscovery(msg)
    msg[1] == ClientID
      reply = '0,' + ServerID
      xmpp_send(ClientID,reply)

def NCAPTIMDiscovery(msg)
    if msg[1] == ClientID
      reply = '0,' + NumTIM + ',' + TIMID
      xmpp_send(ClientID,reply)
```

Transducer Access Services:



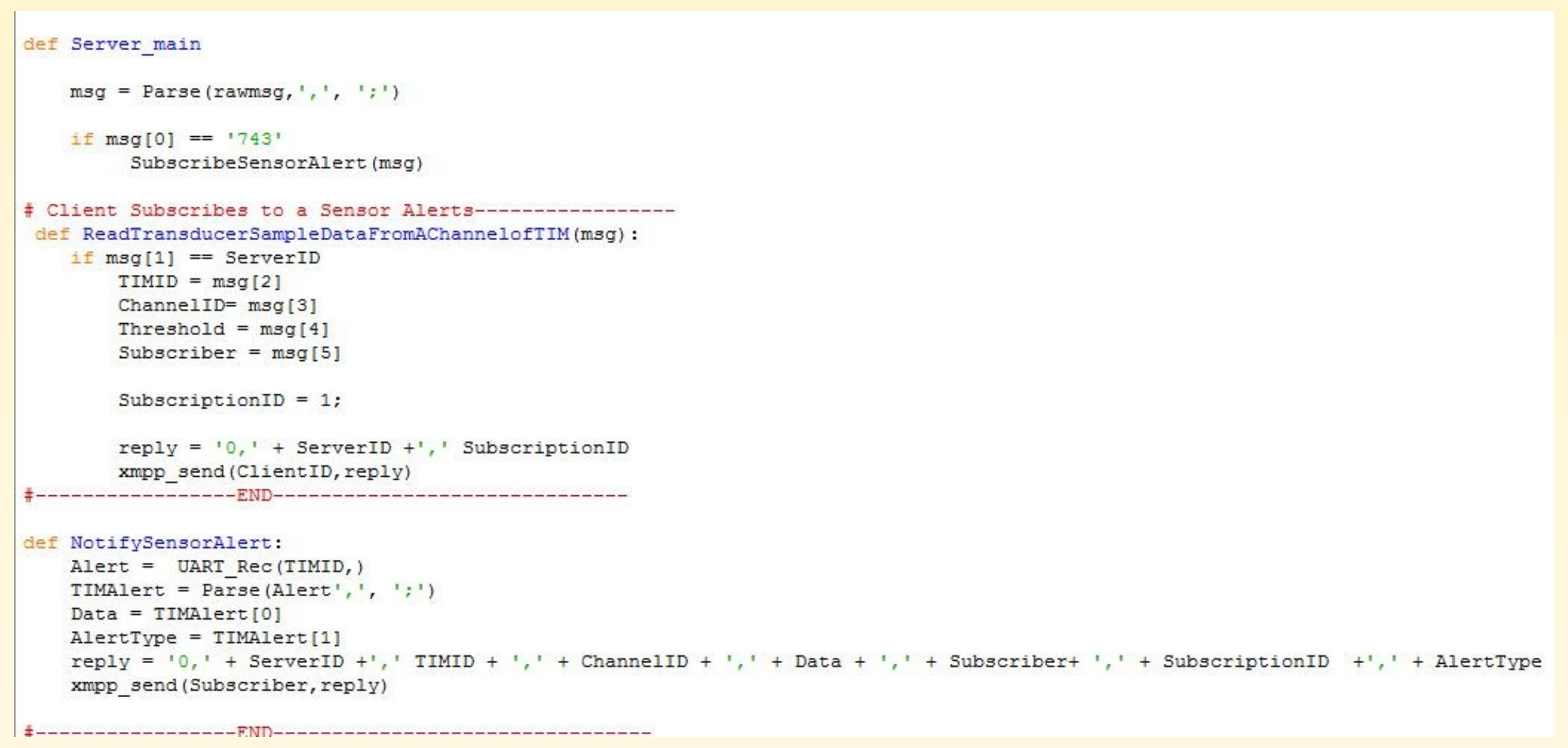
```
def Server_main

    msg = Parse(rawmsg,',', ';')

    if msg[0] == '721'
         ReadTransducerSampleDataFromAChannelofTIM()
    elif msg[0] == '722'
         ReadTransducerBlockDataFromAChannelofTIM()
    elif msg[0] == '723'
         ReadTransducerSampleDataFromMultipleChannelsofTIM()
    elif msg[0] == '724'
         ReadTransducerBlockDataFromMultipleChannelsofTIM()
    elif msg[0] == '725'
         ReadTransducerSampleDataFromMultipleChannelsofMultipleTIM()
    elif msg[0] == '726'
         ReadTransducerBlockDataFromMultipleChannelsofMultiplesTIM()
    elif msg[0] == '727'
         WriteTransducerSampleDataFromMultipleChannelsofTIM()

# Reading Transducer sample data from a single channel of single TIM
 def ReadTransducerSampleDataFromAChannelofTIM(msg)
    if msg[1] == ServerID
        msg[2] = TIMID
        msg[3] = ChannelID
        msg[4] = Timeout

        #Poling TIM for data
        if ChannelID == 1
            UART_send(TIMID,Channel1,'721')
            SampleData =  UART_Rec(TIMID,Channel1);
        elif ChannelID == 2
             UART_send(TIMID,Channel2,'721')
             SampleData =  UART_Rec(TIMID,Channel2);

        reply = '0,' + ServerID +',' TIMID + ',' + ChannelID + ',' SampleData
        xmpp_send(ClientID,reply)
    else
        reply = '1,' + ServerID
        xmpp_send(ClientID,reply)

#End
```

TEDs Access Services:

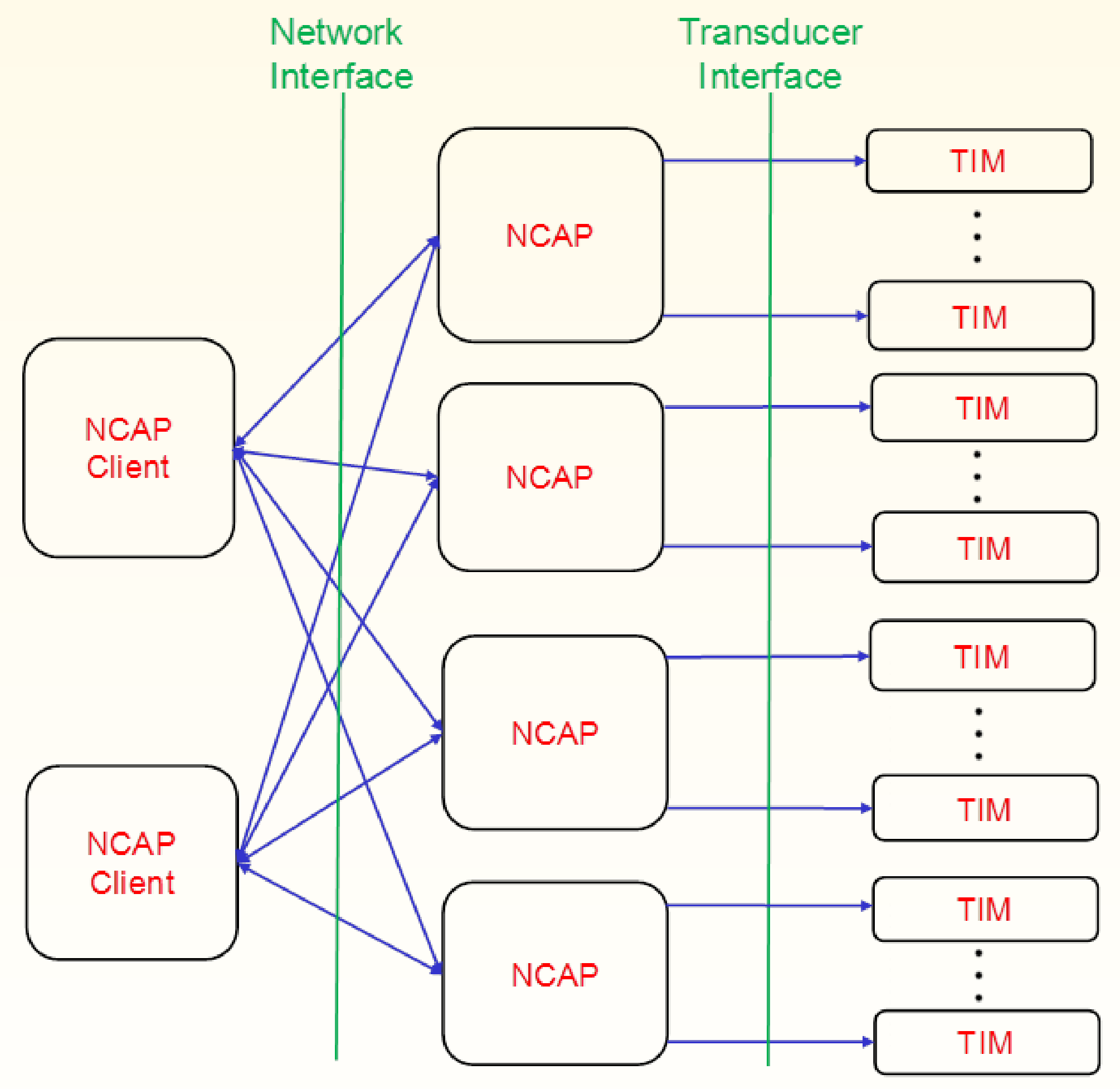


```
def Server_main

    msg = Parse(rawmsg,',', ';')

    if msg[0] == '732'
         ReadTransducerChannelTEDServices()
     elif msg[0] == '7312'
         ReadWriteTransducerChannelTEDSServices()

def ReadTransducerChannelTEDSservices
    if msg[1] == ncapId
        msg[2] = timId
        msg[3] = ChannelId
        msg[4] = Timeout

        #Message to be sent to the TIm
        TimMSG = '732,' + ChannelId

        #Sending request to TIM
         UART_send(timId,ChannelId,TimMSG)
         TEDS1 =  UART_Rec(timId,ChannelId) # Receives the TED information

        reply = '0', +TEDS1
        xmpp_send(ClientID,reply)
    else
        reply = '1,' + ServerID
        xmpp_send(ClientID,reply)
```

Event Notification Services:



```
def Server_main

    msg = Parse(rawmsg,',', ';')

    if msg[0] == '743'
         SubscribeSensorAlert(msg)

# Client Subscribes to a Sensor Alerts-----------------
 def ReadTransducerSampleDataFromAChannelofTIM(msg):
    if msg[1] == ServerID
        TIMID = msg[2]
        ChannelID= msg[3]
        Threshold = msg[4]
        Subscriber = msg[5]

        SubscriptionID = 1;

        reply = '0,' + ServerID +',' SubscriptionID
        xmpp_send(ClientID,reply)
#----------------END------------------------------

def NotifySensorAlert:
    Alert =  UART_Rec(TIMID,)
    TIMAlert = Parse(Alert',', ';')
    Data = TIMAlert[0]
    AlertType = TIMAlert[1]
    reply = '0,' + ServerID +',' TIMID + ',' + ChannelID + ',' + Data + ',' + Subscriber+ ',' + SubscriptionID  +',' + AlertType
    xmpp_send(Subscriber,reply)

#----------------END-------------------------------
```



Future Work

- **Write code for more functions in the Event Notification and Transducer Management Services.**
- **Implement more of the functions that were written into the current model.**
- **Increase the amount of NCAP Servers and Transducer Interface Modules in the model.**

References

[1] IEC/ISO/IEEE P21451-1 "Draft Standard for a Smart Transducer Interface for Sensors and Actuators—Common Network Services"

[2] IEC/ISO/IEEE P21451-1-4 "Information technology — Smart transducer interface for sensors, actuators, and devices — Part 1-4: eXtensible Messaging and Presence Protocol (XMPP) for network device communications"