DIY Captor v1.1

(Arduino Yun with PoE and 5xO3 sensors + 1 Humidity/temperature)

Needed tools:

- **Hair dryer, Hot Air Gun or any other source (e.g., lighter) to heat the heat shrink tubes.**
- **2cm hole Saw with centring drill to making the sensor holes on the waterproof box.**
- **Insulating tape.**
- **Soldering iron and welding wire.**

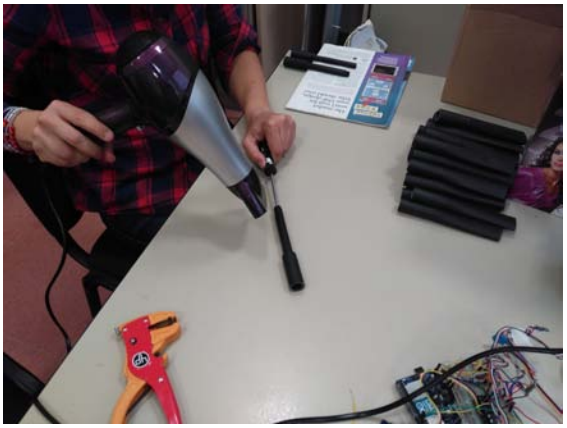
Sensor tubes:

We will start building the sensor tubes:

1. Cut the 1cm, 2cm and heat shrinking tubes into 3 parts of 10, 5 and 15cm respectively. You will need 6 tubes (5 for the O3 sensors and 1 more for the Temp + Humid sensor)

2. Put both PVC tubes inside the heat shrinking tube, the smaller some mm inside the bigger one.

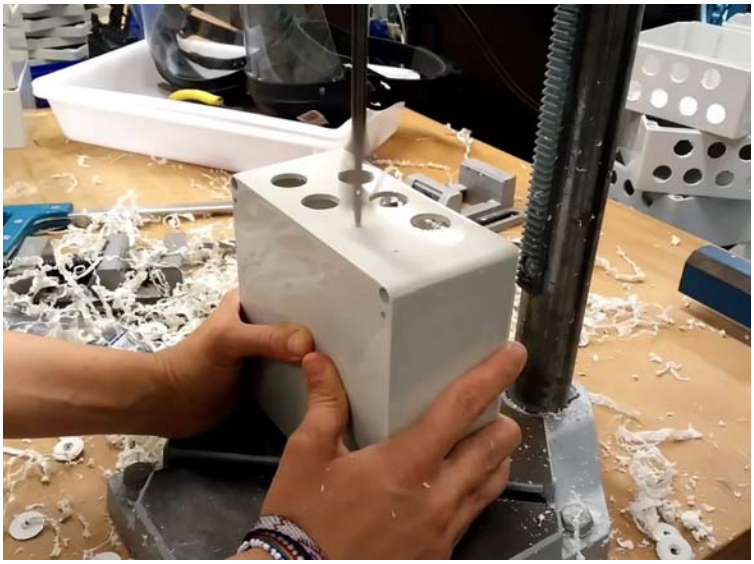
3. Use the hot air gun or hair dryer to heat the tube and make it shrink to the size of the smaller tube. Stretching the tube while it's still hot will help keeping it tighter.




4. Put the cable glands on the tubes and the sensor tubes will be finished.

5. Our box without the cover is 7cm high. With a pencil and a ruler, make a line at 2cm from the top, and another one at 2.5cm from the bottom. This line is where our holes will be centered. as the holes are 2cm diameter, they will be 1/1.5cm from the top and bottom limits.
6. At the sides we need to keep in mind the screws: we will leave a certain margin, and we can place the holes with a distance of 3.5 cm from each other. 3 holes on the upper line and four on the lower line.

7. Use the drill with the 2cm hole saw to make the 7 holes in the box. Also a 0.5 cm hole can be made in the box side to put a LED in it.



8. Clean the holes and put the 6 sensor tubes on them and the power cable gland, tightening them. To be able of tightening the power cable on the gland, you will need to add some extra insulating tape around the cable so that the cable diameter is incremented. Let's work now on the main course :)

Soldering and preparing sensor cables:

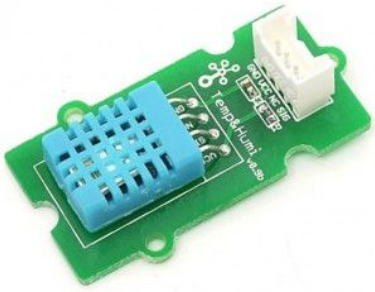
9. Plug the soldering iron, and while it gets warm, let's get the sensors ready for the soldering.

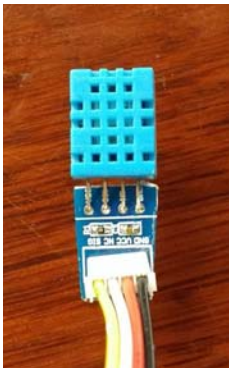
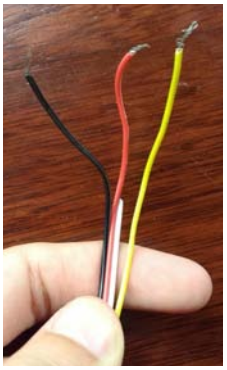
10. To make the MICS-2610 smaller and fit in the tubes, we will desolder the small pins below it, replacing them with coloured long wires (around 30 cm each).
11. Looking at the sensor as it's shown in the following picture, the bottom and top pins are from the "sensor's heater", while the other 2 pins are the "variable sensor resistance". If you want to follow our color code, we used:

RED wire at the TOP (Vcc)
ORANGE at the BOTTOM (R_{heater} = 220 Ohms || 220 Ohms || 330 Ohms)
BROWN at the LEFT (R_{load} = 1kOhm)
RED at the RIGHT (Vcc)

12. We can insert the sensor with its new wires inside the sensor tubes, keeping them inside the tube so that no damage can occur on them.

13. Respect to the humidity and temperature sensor, we had to remove some leftover shield to make it smaller. The 4-cable wire included with the sensor must be modified in order to be able to connect it to the circuitry: we will cut out one of the side connectors and left the four separated cables stripped on its first centimeter or so.

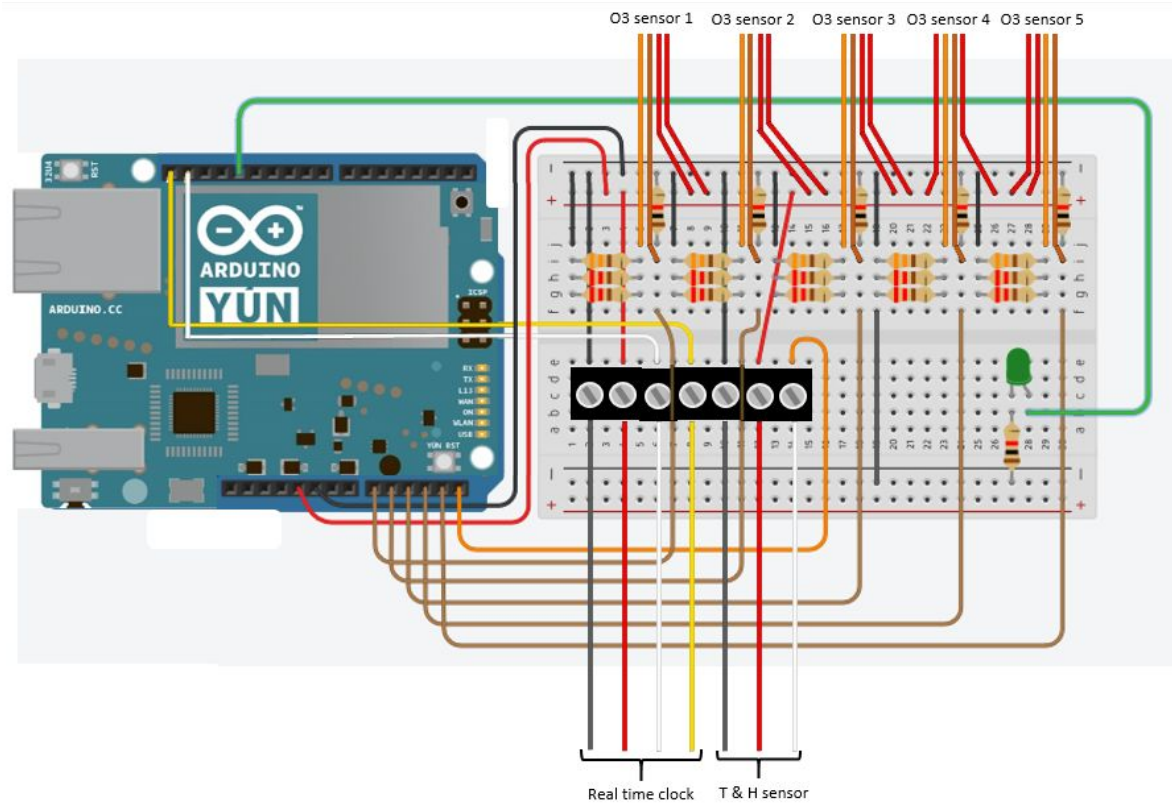


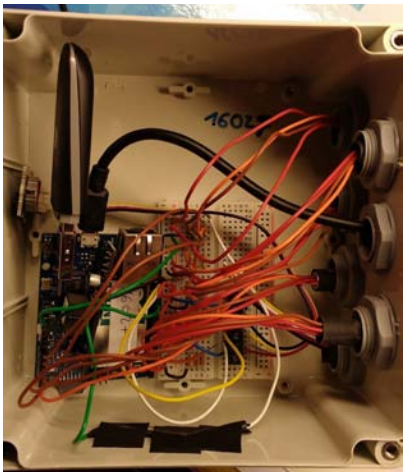
Breadbord sensor circuit:

14. The next step is to build up the sensor circuitry: for the sake of simplicity, we can simply make it with a small breadboard and common electrical components.

15. The breadboard will contain basically a couple of resistances for each sensor, 7 wire-to-breadboard connectors for the Real Time Clock and the Temp and Humi sensor, and the load resistance for the control LED (of course, the LED itself will be fixed in the box hole previously prepared for it, not in the breadboard):

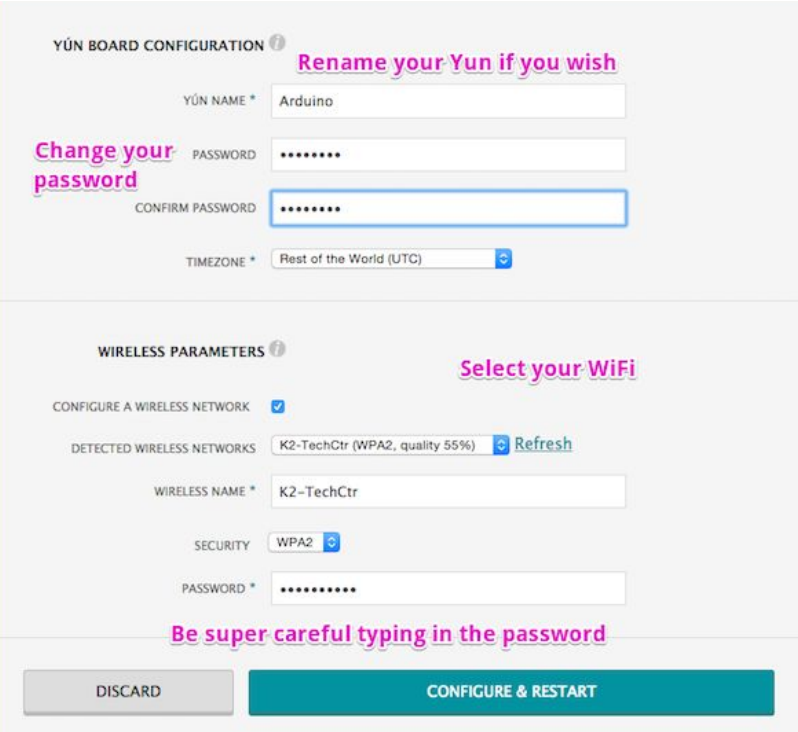


16. One of the RTC 4-cable wire connector must be also cut out and stripped as we have done with the Temp and Humi sensor in order to connect it to the Arduino Yun through the wire-to-breadboard connectors. We will connect also the Temp and Humi sensor, but in this case the white wire can be ignored (just leave it disconnected without stripping it).
17. The Arduino Yun can be now connected to the circuit board, as shown in the schema presented before, and to the power cable.
18. The CR1225 battery can be placed in the RT clock.
19. The Hardware is ready. Now let's go for the software…



Programming the Arduino Yún:

20. First of all, we should check that the firmware installed in the Arduino Yún board is the last available. To do so, go to https://www.arduino.cc/en/Tutorial/YunSysupgrade and follow the instructions, after connecting to the Wi-Fi generated by the Arduino and browsing to 192.168.240.1 (default password is 'doghunter' or 'arduino').
21. Once the Arduino has been upgraded, you can name it (e.g. captorYYSSS, where YY is the last two ciphers of the year of installation and SSS represent the last ciphers of the serial number of the CAPTOR) and set the password.
22. The Arduino can be configured with the WiFi which it will be connected to.



23. As the Linux default system has very little amount of disk space, it is recommended to follow the tutorial on (https://www.arduino.cc/en/Tutorial/ExpandingYunDiskSpace) ExpandingYunDiskSpace to partition the SD card on the Arduino and use one of the partitions as the root system of the Arduino.
24. Using the 'upload_captor.sh' script provided below, we will be able of installing everything we need to have the CAPTOR system on the Arduino Yún board. In a few words, this script copies the different executable files, installs the python packages needed and configures some system parameters. Its parameters are the captor serial numbers (in the form YYSSS commented before) to be configured (they shall be connected to the same network to which our computer is connected). All the files needed can be requested at SANS group, AC Department, UPC (srodrigo@ac.upc.edu).

```
#!/bin/bash
# Batch upload-to-Arduino script, to be run on a computer connected to the Arduino Yún
network
# CAPTOR AC UPC 2016
# @author: srodrigo

if [ $# -eq 3 ]; then
        for i in `seq $1 $2`;
        do
        echo "===================="
        echo $i
        echo "===================="
        sed  's/16000/'$i'/g' captor16000_$3.py > captor$i.py
        sshpass -p "captorcsic" ssh -o StrictHostKeyChecking=no root@captor$i.local
'/etc/init.d/captor stop'
        sshpass -p "captorcsic" ssh -o StrictHostKeyChecking=no root@captor$i.local 'mkdir -p
/bin/captor'
        sed  's/16000/'$i'/g' scp_to_commsensum.sh > scp_to_commsensum_id.sh
        sshpass -p "captorcsic" scp -r -o StrictHostKeyChecking=no ./scp_to_commsensum_id.sh
root@captor$i.local:/root/scp_to_commsensum.sh
        sshpass -p "captorcsic" scp -r ./auto-3g.sh root@captor$i.local:/root/auto-3g.sh
        sshpass -p "captorcsic" scp -r ./ntplib.py root@captor$i.local:/usr/lib/python2.7/
        sshpass -p "captorcsic" scp -r ./captor root@captor$i.local:/etc/init.d/
        sshpass -p "captorcsic" scp -r ./captor$i.py
root@captor$i.local:/bin/captor/captor.py
        sshpass -p "captorcsic" scp -r ./forever.sh root@captor$i.local:/bin/captor/
        sshpass -p "captorcsic" scp -r -o StrictHostKeyChecking=no ./install_captor.sh
root@captor$i.local:/bin/captor/install_captor.sh
        sshpass -p "captorcsic" ssh -o StrictHostKeyChecking=no root@captor$i.local 'chmod
755 /bin/captor/install_captor.sh && /bin/captor/install_captor.sh'
        done
else
        echo "Usage: ./upload captor mincaptorid maxcaptorid number-of-sensors"
        echo -e "\tThis program uploads captorv2 image to the Arduino Yun with\n\tID between
mincaptorid and maxcaptorid, both included"
fi
```

Listing 1.- *Code for installing the CAPTOR system on the Arduino Yún, supposing that the password by default is 'captorcsic'*

```
#!/bin/ash
# Batch install script, to be run on the Arduino Yún
# CAPTOR AC UPC 2016
# @author: srodrigo

# files captor, forever.sh and captor.py must be already copied in /etc/init.d and
/bin/captor, respectively

# Installation of required python packages
opkg update
opkg install distribute
opkg install python-openssl
easy_install pip
pip install requests

# Providing permissions to executable files
```

```
chmod 755 /etc/init.d/captor
chmod 755 /bin/captor/captor.py
chmod 755 /bin/captor/forever.sh
chmod 755 /root/scp_to_commsensum.sh
chmod 755 /root/auto-3g.sh

# Generating secret for authentication purposes if needed
mkdir -p ~/.ssh
dropbearkey -t rsa -f ~/.ssh/id_rsa
dropbearkey -y -f ~/.ssh/id_rsa | grep "^ssh-rsa" >> authorized_keys

# Generating crontab with recurrent tasks
echo "0 1 * * * touch /etc/banner && reboot" >> /etc/crontabs/root
# copy logs every day if needed
echo "0 11 * * * touch /etc/banner && /root/scp_to_commsensum.sh" >> /etc/crontabs/root
# execute tasks if there are any
echo "5 11 * * * touch /etc/banner && /root/today_commands.sh" >> /etc/crontabs/root
# 3G connection daemon
echo "10 * * * * touch /etc/banner && /root/auto-3g.sh" >> /etc/crontabs/root

# Enabling CAPTOR daemon
/etc/init.d/cron enable
/etc/init.d/cron restart
/etc/init.d/captor enable
/etc/init.d/captor restart

# 3G required packages installation
opkg install kmod-usb-serial-option kmod-usb-serial-wwan luci-proto-3g usb-modeswitch-data
usb-modeswitch
```

Listing 2.- *Code to be automatically run in the Arduino to complete the preparation of the CAPTOR system*

25. To check that everything is OK, log in to the Arduino Yún through ssh and check that the captor process is running by executing the command ‘ps | grep captor’, which should have the following result:
26. If 3G is going to be used, it should be configured as follows:
    a. Connect to the Yun throught ssh
    b. open file /etc/usb_modeswitch.d/12d1\:1f01 and copy the following data

```
captorYYSSS: ~# vim /etc/usb_modeswitch.d/12d1\:1f01
########################################################################################
# Huawei E303

TargetVendor=  0x12d1
TargetProduct= 0x14dc

MessageContent="5553424312345678000000000000001106300000010001000000000000000"
########################################################################
```

    c. Disconnect (if it was connected) and/or reconnect the USB 3G modem
    d. Check it's correctly seen by the Yun by typing:

```
captor3: ~# lsusb
... # Example, some data may vary, except the underlined text
Bus 001 Device 007: ID 12d1:1001 Huawei Technologies Co., Ltd. E169/E620/E800 HSDPA
Modem
```

e. Login into the Yun web interface and navigate to the Advanced configuration panel.
f. Navigate to the Network–>Interfaces tab. Click on Add new interface
g. Give the device the name: 3gHuawei
h. Select the interface protocol as: UMTS/GPRS/EV-DO and click Accept
i. Select the following values (or the ones corresponding to your provider):
    i. Modem device: /dev/ttyUSB0
    ii. Service Type: UMTS/GPRS
    iii. APN: movistar.es (this depends on the SIM you are using)
    iv. PIN : ????? (this depends on the SIM you are using)
    v. PAP/CHAP username: MOVISTAR
    vi. PAP/CHAP password: MOVISTAR
j. and click on Save&Apply

27. The rest of the code is provided in the following listings. To solve any doubt, please contact with its author, srodrigo@ac.upc.edu:

```
#!/usr/bin/python
# CAPTOR main process
# CAPTOR AC UPC 2016
# @author: srodrigo
# -*- coding: utf-8 -*-
from subprocess import Popen, PIPE
import subprocess
import datetime
import time
import os
import httplib
import ntplib # Got version 0.3.3 from https://pypi.python.org/pypi/ntplib/
import requests # Installed through pip: (pip install requests)
# To install pip on the Yun:
       # ~:opkg update
       # ~:opkg install distribute
       # ~:opkg install python-openssl
       # ~:easy_install pip
import urllib
import json

# CAPTOR unit name
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
CAPTOR_ID=16000              # CHANGE THIS FOR EVERY CAPTOR!!!!!!!!!!!!!!!!!!!!!!!
CAPTOR_NAME="captor%d" %(CAPTOR_ID) # CHANGE THIS FOR EVERY CAPTOR!!!!!!!!!!!!!!
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
DEBUG = False
# URL allowing access to the Arduino "server" functions via Bridge
```

```
ARDUINO_URL = "localhost/arduino/"
# Arduino's returning date format
DATEFMT = "%Y-%m-%dT%H:%M:%SZ"
# CommSensum date format
DATESTRFMT = "%04d-%02d-%02dT%02d:%02d:%02d"
# Datafile template
FILENAMETEMPLATE = "/mnt/sda1/DATA%04d.txt"
# Data pending to be sent files
FILENAMEPENDING_1 = "/mnt/sda1/pending_1.tmp"
FILENAMEPENDING_2 = "/mnt/sda1/pending_2.tmp"
FILENAMETEMPPENDING = "/mnt/sda1/temppending.tmp"
# Log file
FILENAMELOG = "/mnt/sda1/log.txt"
# Status recovery file
FILENAMESTATUS = "/mnt/sda1/status.txt"
# Time between measures to be sent
SAMPLING_PERIOD = 30

CS_API_KEY = "Rrme1YvMeyi3Ftd69qc4Y2CHk2Yl4L06olUJaWdVVbM"
CS_USER_NAME = "otrullols"
CS_PROJECT_NAME = "CAPTOR"
CS_HTTP_ADDR_1 = "commsensum.pc.ac.upc.edu:3000" # UPC VM
CS_HTTP_ADDR_2 = "51.255.135.164:3000" # DEDICATED SERVER (REPLICATION)
CS_HEADER_CONTENT_TYPE = "application/json"

# Perform a call to an Arduino "server function", returning the output obtained
def arduinoCall(action):
        try:
        p = Popen(["curl", ARDUINO_URL + action], stdin=PIPE, stdout=PIPE, stderr=PIPE)
        output, err = p.communicate()
        rc = p.returncode
        if rc != 0: # Something went bad in the curl process
                return 1
        output = output.translate(None, "\r\n")
        return output
        except Exception as e:
        print "Something went wrong with Arduino call: "
        print e
        print e.message
        return 1

def createSDFileAsTemplate(template):
        i = 0
        while os.path.exists(template %(i)):
        i += 1
        filename = template %(i)
        return filename

def storeMeasurements(dtime,s1,s2,s3,temp,humid,filename):
        try:
        datafile = open(filename,"a")
        line = "%s\t%.04f\t%.04f\t%.04f\t%.02f\t%.02f\n" %(dtime,s1,s2,s3,temp,humid)
        datafile.write(line)
        datafile.close()
        return 0
        except Exception as e:
        print "Couldn't store measurements: "
        print e
        print e.message
        return 1
```

```
def sendPending(filename, url):
        try:
        pendingfile = open(filename, "r")
        lines = pendingfile.readlines()
        pendingfile.close()
        pendingfile = open(FILENAMETEMPPENDING, "w")
        error = 0
        for l in lines:
                err = popPending(url, l)
                if err != 0: # We have not been able of sending it
                error = error + 1
                if err != 2: # The line is correctly formatted
                        pendingfile.write(l)
        pendingfile.close()
        os.remove(filename)
        os.rename(FILENAMETEMPPENDING, filename)
        return error
        except Exception as e:
        print "Error when send pending data: "
        print e
        print e.message
        return -1

def popPending(url, line):
        try:
        data = line.split(";") # Date;Ozone1;Ozone2;Ozone3;Temp;Humidity
        if len(data) < 6:
                return 2 # Parsing error
        dtime = data[0]
        o31 = float(data[1])
        o32 = float(data[2])
        o33 = float(data[3])
        temp = float(data[4])
        hum = float(data[5])
        return sendData(url, dtime, o31, o32, o33, temp, hum) # Redirect error code
        except Exception as e:
        print "Found some corrupted pending data: "
        print e
        print e.message
        return 2 # Parsing error

def pushPending(filename, dtime, s1, s2, s3, t, h):
        try:
        line = "%s;%f;%f;%f;%f;%f\n" %(dtime, s1, s2, s3, t, h)
        pendingfile = open(filename,"a")
        pendingfile.write(line)
        pendingfile.close()
        return 0
        except Exception as e:
        print "Couldn't store pending data: "
        print e
        print e.message
        return 1

def internet_available(url):
        conn = httplib.HTTPConnection(url)
        try:
        conn.request("HEAD", "/")
        return True
```

```
        except Exception as e:
        return False

def sendData(url, dtime, s1, s2, s3, t, h):
        headers = {"user":CS_USER_NAME ,"X-ApiKey": CS_API_KEY,
"Content-type":CS_HEADER_CONTENT_TYPE}

        # Sensor 1
        dataJSON = createBody("O3r",dtime,"%.04f"%(s1),"KOhms")
        r = makePost(createUrl(url,CAPTOR_NAME+"01"),dataJSON,headers)
        if r != 0: # Retry
        r = makePost(createUrl(url,CAPTOR_NAME+"01"),dataJSON,headers)
        if r != 0: # Error
        return r

        # Sensor 2
        dataJSON = createBody("O3r",dtime,"%.04f"%(s2),"KOhms")
        r = makePost(createUrl(url,CAPTOR_NAME+"02"),dataJSON,headers)
        if r != 0: # Retry
        r = makePost(createUrl(url,CAPTOR_NAME+"02"),dataJSON,headers)
        if r != 0: # Error
        return r

        # Sensor 3
        dataJSON = createBody("O3r",dtime,"%.04f"%(s3),"KOhms")
        r = makePost(createUrl(url,CAPTOR_NAME+"03"),dataJSON,headers)
        if r != 0: # Retry
        r = makePost(createUrl(url,CAPTOR_NAME+"03"),dataJSON,headers)
        if r != 0: # Error
        return r

        # Temperature
        dataJSON = createBody("temperature",dtime,"%.02f"%(t),"ºC")
        r = makePost(createUrl(url,CAPTOR_NAME+"temp"),dataJSON,headers)
        if r != 0: # Retry
        r = makePost(createUrl(url,CAPTOR_NAME+"temp"),dataJSON,headers)
        if r != 0: # Error
        return r

        # Humidity
        dataJSON = createBody("humidity",dtime,"%.02f"%(h),"%")
        r = makePost(createUrl(url,CAPTOR_NAME+"humi"),dataJSON,headers)
        if r != 0: # Retry
        r = makePost(createUrl(url,CAPTOR_NAME+"humi"),dataJSON,headers)
        if r != 0: # Error
        return r

        return 0

def createBody(magnitude, dtime, value, unit):
        r = json.dumps({"date":dtime, "magnitude":magnitude, "value":value, "unit":unit})
        r = r.encode('utf-8')
        return r

def createUrl(url,stream):
        updatedUrl = "http://%(hostname)s/v1/%(project)s/%(sname)s" % {"hostname":url,
"project":CS_PROJECT_NAME, "sname":stream}
        if DEBUG == True:
        print "Url created: ", updatedUrl
        return updatedUrl
```

```
def makePost(url, dataJSON, headers):
        if DEBUG == True:
        print url, "-",dataJSON,"-",headers
        try:
        response = requests.post(url, data=dataJSON, headers=headers)

        if response.status_code!=requests.codes.ok:
                logfile.write(response)
                logfile.write(response.text)
                return 1
        else:
                logfile.write("OK\n")
                return 0

        except Exception as e:
        print "Exception on POST: "
        print e
        print e.message
        return 1

# Opening/creating log file
logfile = open(FILENAMELOG, "a")
logfile.write("================================\n")
logfile.write("Hello there. Here we are again.\n")
logfile.write("================================\n")
# Let's begin with some blinks
r = arduinoCall("blink/5")
if r == 1:
        logfile.write("Couldn't perform init blink\n")
        subprocess.call(["reset-mcu"])
        time.sleep(15)
# Now we will create the sequentially numbered datafile
datafile = createSDFileAsTemplate(FILENAMETEMPLATE)
logfile.write("We will write into datafile %s\n" % (datafile))
# Creating (if not existing already) pending file
pendingfile = open(FILENAMEPENDING_1, "a")
pendingfile.close()
pendingfile = open(FILENAMEPENDING_1, "r")
pending_1 = len(pendingfile.readlines())
pendingfile.close()

pendingfile = open(FILENAMEPENDING_2, "a")
pendingfile.close()
pendingfile = open(FILENAMEPENDING_2, "r")
pending_2 = len(pendingfile.readlines())
pendingfile.close()
logfile.write("Pending values are: %d (UPC VM) %d (Dedicated)\n" %(pending_1, pending_2))
logfile.close()

logfile = open(FILENAMELOG, "a")
# Compare with NTP hour and refresh RTC if possible
if internet_available("0.openwrt.pool.ntp.org"):
        try:
        ntpclient = ntplib.NTPClient()
        ntpd =
datetime.datetime.utcfromtimestamp(ntpclient.request('0.openwrt.pool.ntp.org').tx_time)
        logfile.write("NTP server says it is now %s\n" % (ntpd))
        try:
                output = arduinoCall("setDate/%d/%d/%d/%d/%d/%d" %(ntpd.year, ntpd.month,
```

```
ntpd.day, ntpd.hour, ntpd.minute, ntpd.second))
                if output == 1:
                logfile.write("Couldn't set date\n")
                subprocess.call(["reset-mcu"])
                time.sleep(15)
                else:
                try:
                        rtcd = datetime.datetime.strptime(output,DATEFMT)
                        logfile.write("RTC set correctly to %s\n" % (rtcd))
                except Exception as e:
                        print "No RTC available: "
                        subprocess.call(["reset-mcu"])
                        time.sleep(15)
                        print e
                        print e.message
        except NameError:
                print "Bad ntp datetime"

        except Exception as e:
        print "Exception when syncing with NTP: "
        print e
        print e.message

# We will now initialize the measure variables
meantemp = 0
meanhumid = 0
meano31 = 0
meano32 = 0
meano33 = 0
countmeasures = 0
stacksend = []

logfile.close()

first_measure = True # Send always the first measure to test connection
# Main loop
while (True):
        logfile = open(FILENAMELOG, "a")
        initsec = time.time()
        if pending_1 > 0 and pending_2 > 0:
        r = arduinoCall("blink/1")
        if r == 1:
                logfile.write("Couldn't perform loop blink\n")
                subprocess.call(["reset-mcu"])
                time.sleep(15)
        else:
        r = arduinoCall("blink/3")
        if r == 1:
                logfile.write("Couldn't perform loop blink\n")
                subprocess.call(["reset-mcu"])
                time.sleep(15)

        # Let's get Date and Time
        # Bad clock datetime: 2165-165-165T165:165:85Z

        dtime = arduinoCall("getDate")
        if dtime == 1:
        logfile.write("Couldn't get date from RTC\n")
        subprocess.call(["reset-mcu"])
        time.sleep(15)
```

```
        rtcd = datetime.datetime.now()
        else:
        try:
                rtcd = datetime.datetime.strptime(dtime,DATEFMT)
        except ValueError:
                rtcd = datetime.datetime.now()

        logfile.write("Now it is %s\n" % (rtcd))
        output = arduinoCall("getSensors")
        if output == 1:
        logfile.write("Couldn't get sensors data\n")
        subprocess.call(["reset-mcu"])
        time.sleep(15)
        else:
        try:
                [temp, humid, o31, o32, o33] = output.split("\t")
                if DEBUG == True:
                logfile.write("Sensor 1 value is " + o31 + "\n")
                logfile.write("Sensor 2 value is " + o32 + "\n")
                logfile.write("Sensor 3 value is " + o33 + "\n")
                logfile.write("Temp value is " + temp + "\n")
                logfile.write("Humidity value is " + humid + "\n")
                meano31 += float(o31)
                meano32 += float(o32)
                meano33 += float(o33)
                meantemp += float(temp)
                meanhumid += float(humid)
                countmeasures += 1
                print stacksend
                # Let's see if it's time to upload data
                # Each captor uploads data in a ID related instant of time, each 30 min
                if rtcd.minute%SAMPLING_PERIOD == 0 or countmeasures >= SAMPLING_PERIOD or
first_measure == True:
                if countmeasures != 1 or first_measure == True:
                        logfile.write("It is time to take measures\n")
                        meano31 /= countmeasures
                        meano32 /= countmeasures
                        meano33 /= countmeasures
                        meantemp /= countmeasures
                        meanhumid /= countmeasures
                        rtcstr = DATESTRFMT %(rtcd.year, rtcd.month, rtcd.day, rtcd.hour,
rtcd.minute, rtcd.second)
                        r =
storeMeasurements(rtcstr,meano31,meano32,meano33,meantemp,meanhumid,datafile)
                        listtemp = []
                        print stacksend
                        listtemp.append(meanhumid)
                        listtemp.append(meantemp)
                        listtemp.append(meano33)
                        listtemp.append(meano32)
                        listtemp.append(meano31)
                        listtemp.append(rtcstr)
                        stacksend.append(listtemp)
                        print stacksend
                        if r != 0:
                        logfile.write("Couldn't store measurements\n")

                meantemp = 0
                meanhumid = 0
                meano31 = 0
```

```
                meano32 = 0
                meano33 = 0
                countmeasures = 0

                if ((rtcd.minute - (CAPTOR_ID%100))%SAMPLING_PERIOD == 0 or first_measure ==
True) and stacksend != []:
                first_measure = False
                logfile.write("It is time to communicate\n")
                for measure in stacksend:
                        dt = measure.pop()
                        ozone1 = measure.pop()
                        ozone2 = measure.pop()
                        ozone3 = measure.pop()
                        te = measure.pop()
                        hu = measure.pop()
                        # CS_HTTP_ADDR_1 (UPC VM)
                        err = sendData(CS_HTTP_ADDR_1, dt, ozone1, ozone2, ozone3, te, hu)

                        if (err != 0):
                        r = pushPending(FILENAMEPENDING_1, dt, ozone1, ozone2, ozone3, te, hu)
                        if r != 0:
                                logfile.write("Couldn't store pending data (UPC VM)\n")
                        else:
                        temp = sendPending(FILENAMEPENDING_1,CS_HTTP_ADDR_1)
                        if temp != -1:
                                pending_1 = temp

                        # CS_HTTP_ADDR_2 (DEDICATED)
                        err = sendData(CS_HTTP_ADDR_2, dt, ozone1, ozone2, ozone3, te, hu)

                        if (err != 0):
                        r = pushPending(FILENAMEPENDING_2, dt, ozone1, ozone2, ozone3, te, hu)
                        if r != 0:
                                logfile.write("Couldn't store pending data (DEDICATED)\n")
                        else:
                        temp = sendPending(FILENAMEPENDING_2,CS_HTTP_ADDR_2)
                        if temp != -1:
                                pending_1 = temp

                stacksend = []
        except Exception as e:
                logfile.write("Error when taking data\n")
                subprocess.call(["reset-mcu"])
                time.sleep(15)
                print "No data was token: "
                print e
                print e.message

        #We will sleep for the rest of the minute
        sleeptime = 60-(time.time()-initsec)
        if sleeptime < 0:
        sleeptime = 0
        time.sleep(sleeptime)
        logfile.close()
```

Listing 3.- *Code of the main CAPTOR process, in this case for three sensors onboard*

```
#!/bin/sh
# 3G connectivity maintenance daemon
# CAPTOR AC UPC 2016
# @author: srodrigo

# This script checks the 3G connectivity and tries to recover it in case it is
# not working properly. If so, it saves the logread in a SD card file
# (for debug purposes). Always logs the status and the actions performed.

# Let's first obtain the date and hour right now
DATE=`date +%Y-%m-%d\|%H\:%M\:%S`

# Now we are going to check for the "logs" folder
# Due to possible problems in the Linino with mounting devices (the 3G USB
# sometimes messes up with the SD), we will first look for any SD mounted

ls /mnt/sda1 > /dev/null

if [ $? -ne 0 ]
then # There is no sd, let's save log in /root
    LOGFILE="/root/3G-connection-log.txt"
else # SD is mounted on /mnt/sda1
    LOGFILE="/mnt/sda1/3G-connection-log.txt"
fi

# Printing a general connectivity status message
echo "" >> $LOGFILE
echo "" >> $LOGFILE
echo "\|||||||||||||||||||||||||||||||||||||||/" >> $LOGFILE
echo "-        3G CONNECTION LOG             -" >> $LOGFILE
echo -n "-     DATE: " >> $LOGFILE
echo -n "$DATE" >> $LOGFILE
echo "         -" >> $LOGFILE
echo "/|||||||||||||||||||||||||||||||||||||||\\" >> $LOGFILE
echo "" >> $LOGFILE
echo "===============" >> $LOGFILE
echo "IFCONFIG STATUS" >> $LOGFILE
echo "===============" >> $LOGFILE
ifconfig >> $LOGFILE
echo "" >> $LOGFILE
echo "======" >> $LOGFILE
echo "ROUTES" >> $LOGFILE
echo "======" >> $LOGFILE
route -n >> $LOGFILE

# Let's check the connectivity to Internet
echo "" >> $LOGFILE
echo "==========" >> $LOGFILE
echo "PING CHECK" >> $LOGFILE
echo "==========" >> $LOGFILE
ping -q -w 10 www.google.es >> $LOGFILE
if [ $? -ne 0 ]
then
    echo "" >> $LOGFILE
    echo "======================" >> $LOGFILE
    echo "No Internet connection" >> $LOGFILE
    echo "======================" >> $LOGFILE
    CONNECTED=0
else
    echo "" >> $LOGFILE
```

```
    echo "==================" >> $LOGFILE
    echo "3G IFACE PING CHECK" >> $LOGFILE
    echo "==================" >> $LOGFILE
    ping -q -I 3g-3gHuawei -w 10 www.google.es  >> $LOGFILE
    if [ $? -ne 0 ]
    then
    echo "" >> $LOGFILE
       echo "======================================" >> $LOGFILE
       echo "Connected to Internet via other ifaces" >> $LOGFILE
       echo "======================================" >> $LOGFILE
        CONNECTED=2
    else
        echo "" >> $LOGFILE
        echo "============================" >> $LOGFILE
        echo "Connected to Internet via 3G" >> $LOGFILE
        echo "============================" >> $LOGFILE
        CONNECTED=1
    fi
fi

if [ $CONNECTED -ne 1 ]
then
    ifconfig | grep -q "3g-3gHuawei"
    if [ $? -eq 0 ]
    then
        echo "" >> $LOGFILE
        echo "============================================================" >> $LOGFILE
        echo "It seems that the interface is up. Printing system log..." >> $LOGFILE
        echo "============================================================" >> $LOGFILE
        logread >> $LOGFILE
        ifconfig 3g-3gHuawei down
        ifconfig 3g-3gHuawei up
        route add default gw 10.64.64.64 3g-3gHuawei
        ping -q -I 3g-3gHuawei -w 10 www.google.es
       if [ $? -ne 0 ]
        then
               echo "" >> $LOGFILE
        echo "====================================" >> $LOGFILE
              echo "3G interface restarted without result" >> $LOGFILE
               echo "====================================" >> $LOGFILE
               if [ $CONNECTED -ne 1 ] # 3G dongle is down; let's reconnect it
               then
                      echo "" >> $LOGFILE
                      echo "======================" >> $LOGFILE
                      echo "Rebooting 3G dongle..." >> $LOGFILE
                      echo "======================" >> $LOGFILE
                      echo 0 > /sys/bus/usb/devices/1-1.1/authorized
                      sleep 1
                      echo 1 > /sys/bus/usb/devices/1-1.1/authorized
                      sleep 5
                      /etc/init.d/network restart
                      sleep 15
                      route add default gw 10.64.64.64 3g-3gHuawei
                      ping -q -I 3g-3gHuawei -w 10 www.google.es
                      if [ $? -ne 0 ]
                      then
                             echo "" >> $LOGFILE
                             echo "================================" >> $LOGFILE
                             echo "3G dongle rebooted without result" >> $LOGFILE
                             echo "================================" >> $LOGFILE
```

```
                        else
                                echo "" >> $LOGFILE
                                echo "=====================================================" >>
$LOGFILE
                                echo "3G dongle rebooted. Internet connection restablished" >>
$LOGFILE
                                echo "=====================================================" >>
$LOGFILE
                        fi
                fi
        else
                echo "" >> $LOGFILE
                echo "=========================================================" >> $LOGFILE
                echo "3G interface restarted. Internet connection restablished" >> $LOGFILE
                echo "=========================================================" >> $LOGFILE
        fi
    else
        echo "" >> $LOGFILE
        echo "=========================================================" >> $LOGFILE
        echo "The 3G interface does not exist. Printing system log..." >> $LOGFILE
        echo "=========================================================" >> $LOGFILE
        logread >> $LOGFILE
        ifconfig 3g-3gHuawei up
        route add default gw 10.64.64.64 3g-3gHuawei
        ping -q -I 3g-3gHuawei -w 10 www.google.es
       if [ $? -ne 0 ]
        then
                echo "" >> $LOGFILE
        echo "====================================" >> $LOGFILE
              echo "3G interface restarted without result" >> $LOGFILE
               echo "====================================" >> $LOGFILE
               if [ $CONNECTED -ne 1 ] # 3G dongle is down; let's reconnect it
               then
                        echo "" >> $LOGFILE
                        echo "======================" >> $LOGFILE
                        echo "Rebooting 3G dongle..." >> $LOGFILE
                        echo "======================" >> $LOGFILE
                        echo 0 > /sys/bus/usb/devices/1-1.1/authorized
                        sleep 1
                        echo 1 > /sys/bus/usb/devices/1-1.1/authorized
                        sleep 5
                        /etc/init.d/network restart
                        sleep 15
                        route add default gw 10.64.64.64 3g-3gHuawei
                        ping -q -I 3g-3gHuawei -w 10 www.google.es
                        if [ $? -ne 0 ]
                        then
                                echo "" >> $LOGFILE
                                echo "================================" >> $LOGFILE
                                echo "3G dongle rebooted without result" >> $LOGFILE
                                echo "================================" >> $LOGFILE
                        else
                                echo "" >> $LOGFILE
                                echo "=====================================================" >>
$LOGFILE
                                echo "3G dongle rebooted. Internet connection restablished" >>
$LOGFILE
                                echo "=====================================================" >>
$LOGFILE
                        fi
```

```
                fi
        else
                echo "" >> $LOGFILE
                echo "=========================================================" >> $LOGFILE
                echo "3G interface restarted. Internet connection restablished" >> $LOGFILE
                echo "=========================================================" >> $LOGFILE
         fi
    fi
fi
```

Listing 4.- *Code of the 3G connectivity maintenance daemon that recovers the connectivity whenever there are network or device problems*

```
#!/bin/sh /etc/rc.common
# Captor init script
# CAPTOR AC UPC 2016
# @author: srodrigo

START=10
STOP=15

start() {
        echo "starting captor service"
        reset-mcu
        sleep 15 # Wait for mcu
        /bin/captor/forever.sh &
        # commands to launch application
}

stop() {
        echo "stopping captor service"
        killall forever.sh
        killall captor.py
        # commands to kill application
}
```

Listing 5.- *CAPTOR daemon init script*

```
#!/bin/ash

# run forever captor python daemon
```

```
while true; do
    /bin/captor/captor.py > /mnt/sda1/errorlog.txt 2> /mnt/sda1/errorlog.txt
    sleep 5
done
```

Listing 6.- *run-forever simple approach for the CAPTOR daemon*