Publicaciones Técnicas
pubtecia@astro.unam.mx

Instituto de Astronomía
UNAM

Comunicación Interna CI-2014-02

Aceptación: 07-03-2014
Recepción: 12-10-2012

Weather_daemon: Monitor de procesos de la estación meteorológica del OAN.

E. Colorado

Instituto de Astronomía. Universidad Nacional Autónoma de México.
Km. 103 Carretera Tijuana-Ensenada, Ensenada, B.C., México.

RESUMEN:

En este trabajo se presenta un método para verificar que los procesos de la estación meteorológica estén funcionando correctamente y generando los datos meteorológicos cada 5 minutos, de lo contrario este monitor restablece todos los procesos para que funcione adecuadamente.

Contenido

Colorado.

1. INTRODUCCIÓN

El Observatorio Astronómico Nacional (OAN) ubicado en San Pedro Mártir (SPM) cuenta con una estación meteorológica modelo "**Davis Vantage Pro2 Plus**" para el monitoreo y registro histórico del clima de la región. Desafortunadamente, las consolas de estos dispositivos resultaron susceptibles al ruido eléctrico inducido por las plantas generadoras y rayos eléctricos, generando una desconexión entre el puerto serie de la consola y la PC.

Debido a lo anterior, la estación meteorológica deja de reportar los parámetros del clima y se tienen que reiniciar todos los procesos y módulos de comunicación serie de forma ordenada. Hasta ahora, este proceso se ha venido realizando de forma manual.

Por esa razón se desarrolló el monitor de datos del clima, el cual, al detectar una falta de datos, reinicializa de forma automática todos los procesos involucrados con la adquisición de los parámetros meteorológicos.

2. MÉTODO

El método está basado en el uso de "daemons" del sistema operativo (OS) Linux, los cuales se ejecutan de forma automática al encender la PC y corren en el fondo de los procesos del OS sin afectar a otros programas.

La estación meteorológica actualiza sus datos cada minuto y por esto, el monitor *weather_daemon* verifica que existan datos nuevos cada minuto; da hasta un plazo máximo de 8 minutos para que los datos se actualicen. Si la diferencia de la fecha de los últimos datos registrados por la estación, comparada con la fecha de la PC, es mayor a 8 minutos, este monitor reinicializará todos los componentes asociados a la adquisición de datos de la estación meteorológica.

Es importante verificar que los datos se actualicen cada minuto y que el sistema de la estación meteorológica funcione correctamente ya que esto nos permitirá cerrar rápidamente los telescopios en caso de un evento meteorológico tal como lluvia, humedad o vientos fuertes.

3. DESAROLLO

Este sistema de monitoreo se desarrolló para el sistema operativo Linux en el lenguaje de programación Python y consta de 3 programas que se describen a continuación. Su código fuente se lista en los apéndices.

3.1 PROGRAMA PRINCIPAL, WEATHER_DAEMON.PY

Éste es el programa principal; se encarga de registrarse en el sistema operativo como un proceso tipo "*daemon*" y ejecuta cada minuto la rutina **checa_clima()** de la clase c_weather_data.py, que se describe en la sección siguiente.

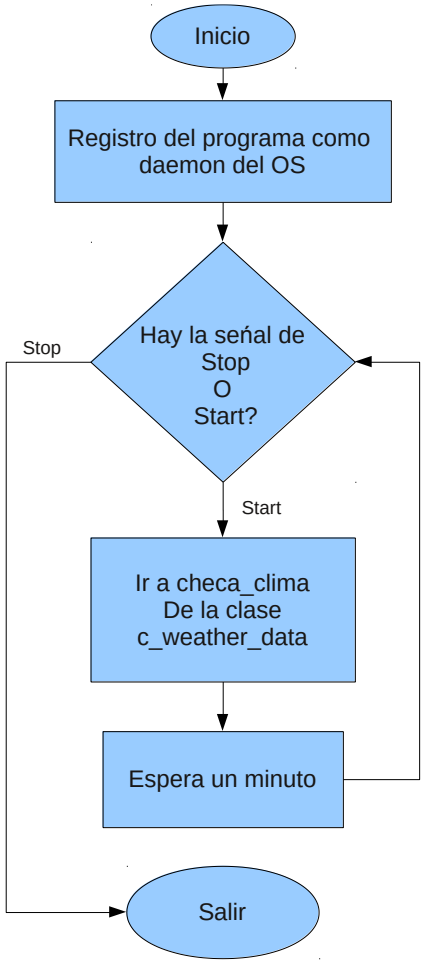

Figura 1: *Diagrama de flujo de weather_daemon.py.*

En la *Figura 1*, se presenta el diagrama de flujo del programa principal weather_daemon.py, el cual puede aceptar como argumento la palabra "**start**" o "**stop**". Al iniciar el programa con el argumento "start", por ejemplo: "weather_daemon.py start", el programa iniciará el proceso de verificación de los datos del clima y liberará inmediatamente el proceso o terminal donde éste se ejecutó ya que se registra como un proceso "daemon" del sistema operativo. Si el programa se ejecuta con el argumento "stop", entonces sólo le dice al sistema operativo que elimine la ejecución del programa.

3.2 CLASE C_WEATHER_DATA.PY

Esta clase se encarga de leer los últimos datos grabados por los programas de adquisición de la estación meteorológica que se encuentran en *"/var/lib/wview/img/Archive*", después compara la diferencia de tiempo contra la fecha de la PC y si la diferencia es mayor que 8 minutos, se ejecuta la rutina **do_hard_restart().**

La rutina **do_hard_restart()** realiza los pasos necesarios para reinicializar todos los procesos relacionados a los programas de adquisición de la estación meteorológica incluyendo el módulo del kernel **cp210x** encargado de la comunicación serial de la consola. Los pasos que se ejecutan son los siguientes:

#	Instrucción	Descripción
1	/etc/init.d/wview stop	Ejecuta el script del software de la estación meteorológica para parar todos los subprogramas asociados a la adquisición. *
2	killall /usr/bin/wviewd_vpro	Elimina el proceso encargado de la adquisición de los datos de la consola Davis Vantage Pro.
3	killall /usr/bin/htmlgend	Elimina el proceso encargado de la generación de la página web.
4	killall /usr/bin/wvcwopd	Elimina el proceso encargado de la transmisión de los datos a CWOP.
5	killall /usr/bin/wvhttpd	Elimina el proceso encargado de enviar los datos a Weather Underground.
6	killall /usr/bin/wviewsshd	Elimina el proceso encargado de transmitir los datos vía ssh a la PC de respaldo.
7	killall /usr/bin/wvpmond	Elimina el proceso encargado de monitorear sus propios autoprocesos.
8	killall /usr/bin/radmrouted	Elimina el proceso encargado de la generación de la gráfica.
9	rm /var/lib/wview/*.pid	Elimina los archivos de candado de software de adquisición.
10	rmmod cp210x	Remueve del kernel el módulo encargado de la comunicación serial hacia la consola Davis Vantage Pro
11	modprobe cp210x	Reinstala en el kernel el módulo encargado de la comunicación serial hacia la consola Davis Vantage Pro
12	sleep 5	Espera 5 segundos a que el módulo cp210x configure la terminal serial.
13	/etc/init.d/wview start	Ejecuta el script del software de la estación meteorológica para iniciar todos los subprogramas asociados a la adquisición.*

* Para mayor información, consulte el manual de usuario del programa wview [2].

El diagrama de flujo de este programa se muestra a continuación.

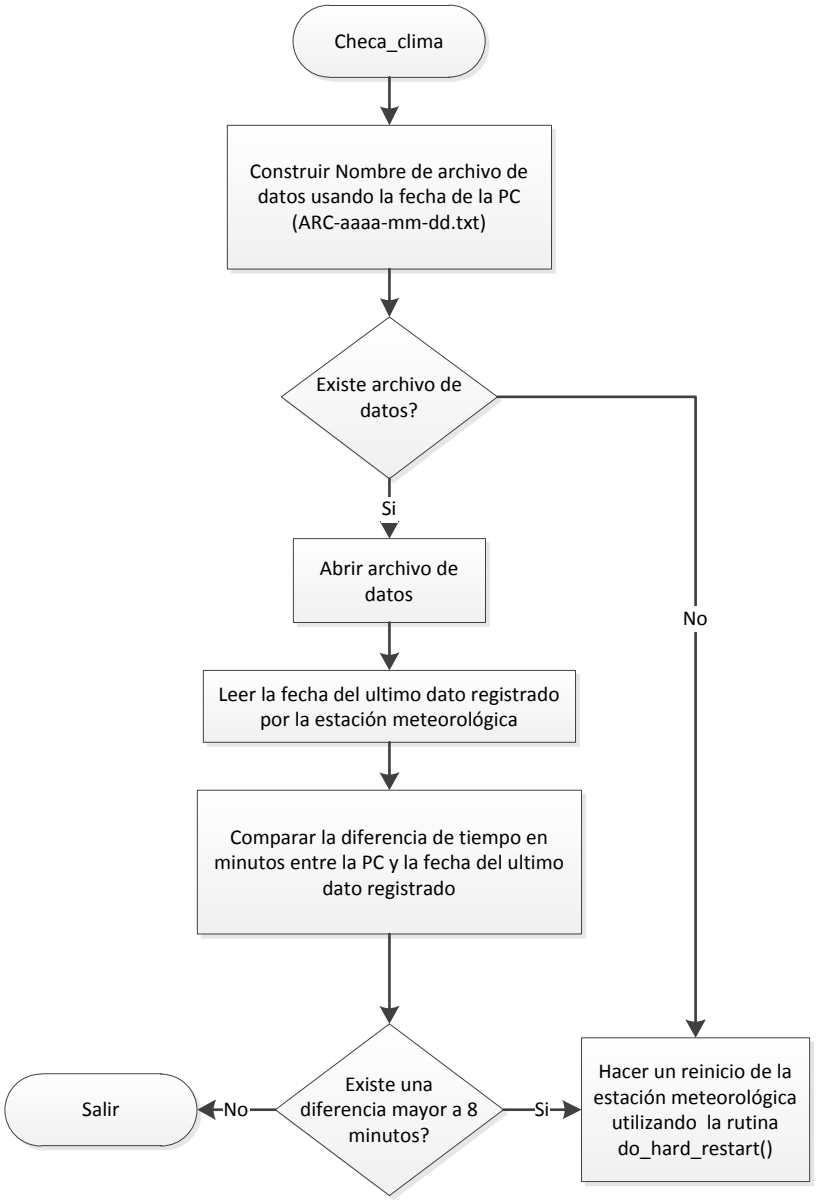

Figura 2: *Diagrama de flujo de la función checa_clima perteneciente a la clase c_weather_data.py.*

Como se puede observar en la *Figura 2*, incluimos la condición de reinicialización de los procesos de la estación meteorológica aun cuando el archivo de datos no exista ya que no habrá forma de saber cuál fue el último dato adquirido; suponemos que hay un problema y reinicializamos todo.

3.3 SCRIPT DE INICIALIZACIÓN MY_DAEMON

Este es el script de inicialización y parada de nuestro programa monitor, el cual se lista en el Apéndice C. Sirve para que el sistema operativo inicie o detenga el programa automáticamente al momento de encender o apagar la PC.

En otras palabras, el script, al iniciar la PC, ejecuta al programa monitor con el argumento **start**, como se ejemplifica en el mando siguiente:

- /root/Daemon_weather/weather_daemon.py start

Al apagar la PC, anexa el argumento **stop**, como se ejemplifica en el mando siguiente:

- /root/Daemon_weather/weather_daemon.py stop

Este script se encuentra localizado en **/etc/init.d**, donde por default se localizan todos los daemons del sistema operativo Linux.

A continuación se muestra el diagrama de flujo del encendido de una PC bajo el OS Linux mostrando, al final, el momento donde carga los deamons.

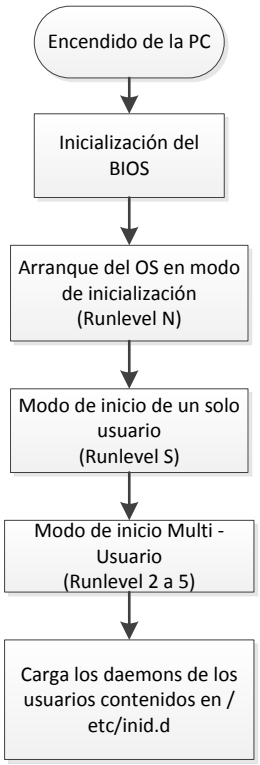

Figura 3: *Proceso de encendido y carga del Daemon.*

Para obtener más información, refiérase a [3].

4. DEPENDENCIA DE BIBLIOTECAS

Este programa fue desarrollado y probado en el sistema operativo Ubuntu 10.04 (Lucid Lynux), Ubuntu 12.04 (Precise Pangolin) y Debian 6.0.5 (squeeze) y requiere una sola biblioteca externa que se instala con el siguiente mando:

- sudo apt-get install python-daemon

5. CONCLUSIONES

Se ha probado con éxito este programa monitor con la estación meteorológica del IA-Ensenada y del OAN-SPM. Anteriormente la estación dejaba de reportar sus datos aproximadamente cada semana y, hasta que alguien reportaba la falla, se corregía manualmente, pero ahora, con este programa monitor, se corrige automáticamente el problema de adquisición teniendo disponibles constantemente los datos meteorológicos.

Esto es muy importante para el telescopio robótico del 1.5m del OAN debido a que sus procesos de apertura se basan en las condiciones reportadas con esta estación meteorológica.

6. REFERENCIAS

[1] E. Colorado, D. Hiriart.
"Estación Meteorológica del Observatorio Astronómico Nacional en San Pedro Mártir, B. C."
Publicaciones Técnicas del Instituto de Astronomía, UNAM.
Comunicación Interna CI-2007-04.
2007.

[2] "Wview User Manual".
http://www.wviewweather.com/release-notes/wview-User-Manual.html

[3] "Manual de referencia de los OS basados en Debian".
http://www.debian.org/doc/manuals/debian-reference/

APÉNDICE A. LISTADO DEL PROGRAMA WEATHER_DAEMON.PY

```
#!/usr/bin/python
'''
Ocupa apt-get install python-daemon

E. Colorado, V0.1 -Sep-2012, Inicio, deamon para wview por bug en crontab con script de tcl
'''
import time
from daemon import runner
import c_weather_data
from c_weather_data import *
################################################################################
class App(Weather_data):
      def __init__(self):
      self.stdin_path = '/dev/null'
      self.stdout_path = '/tmp/w.log'
      self.stderr_path = '/tmp/we.log'
      self.pidfile_path =  '/tmp/weather_daemon.pid'
      self.pidfile_timeout = 5
      Weather_data.__init__(self)
################################################################################
   def run(self):
      while True:
          localtime = time.asctime( time.localtime(time.time()) )
         self.checa_clima()
         time.sleep(60)
################################################################################
app = App()
daemon_runner = runner.DaemonRunner(app)
daemon_runner.do_action()
```

APÉNDICE B. LISTADO DEL PROGRAMA C_WEATHER_DATA.PY

```
#!/usr/bin/python
import time
import os.path
import os
#############################################################################
class Weather_data():
   def __init__(self):
      self.path = '/var/lib/wview/img/Archive' #las demas, default
      self.Maximo_minutos=8.0
#############################################################################
   def lee_archivo(self):
      #construir nombre de archivo
      localtime = time.localtime(time.time())

      Archivo=time.strftime("ARC-%Y-%m-%d.txt", localtime)
      print 'Archivo',Archivo
      fullname = os.path.join(self.path,Archivo)
      print fullname
      if not os.path.exists(fullname):
         print 'No existe archivo',fullname
         return False
      print 'Si existe el archivo, voy a leerlo'

      openfile = open(fullname, 'r')
      str=openfile.read()
      openfile.close()
      lineas=str.split('\n')

      #ultimo dato grabado
      ultima= lineas[-2]

      #construir estructura de fecha
      fecha=ultima[0:14]
      print 'Ultima fecha archivo --->',fecha
      self.fecha_archivo=time.strptime(fecha,"%Y%m%d %H:%M")
      return True
#############################################################################
   def verifica_fecha(self):
      old=0
      localtime = time.localtime(time.time())
      now= time.mktime( localtime )
      otro=time.mktime( self.fecha_archivo)
      old=(now-otro)/60.0
      print 'ha pasado',old,' minutos'
      return old
#############################################################################  def
checa_clima(self):
      ok=self.lee_archivo()
      if not ok:
         self.do_hard_restart()
         return False
```

```
        old=self.verifica_fecha()
        if old >self.Maximo_minutos:
            print 'Ya pasaron ',old,' minutos, Hacer Restart'
            self.do_hard_restart()
            return False
        print 'Todo OK'
        return True
##############################################################################
    def  do_hard_restart(self):
        print 'Haciendo Restart de la estacion'
        mandos=['/etc/init.d/wview stop',
                'killall /usr/bin/wviewd_vpro',
                'killall /usr/bin/htmlgend',
                'killall /usr/bin/wvcwopd',
                'killall /usr/bin/wvhttpd',
                'killall /usr/bin/wviewsshd',
                'killall /usr/bin/wvpmond',
                'killall /usr/bin/radmrouted',
                'rm /var/lib/wview/*.pid',
                'rmmod cp210x',
                'modprobe cp210x',
                'sleep 5',
                '/etc/init.d/wview start',
                ]
        for cmd in mandos:
            print cmd
            os.system(cmd)
```

APÉNDICE C. LISTADO DEL SCRIPT MY_DAEMON

```
#! /bin/sh
# E. Colorado

### BEGIN INIT INFO
# Provides:          my_daemon
# Required-Start:    $local_fs $syslog $time
# Required-Stop:     $local_fs $syslog $time
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start daemon at boot time to check weather data
# Description:       Reboot wview if no data is available, also reinstall serial module
### END INIT INFO

#este script debe estar en:  /etc/init.d/
#actualizar con: update-rc.d my_daemon defaults
#asegurar que tenga bit de ejecuccion: chmod 755 my_daemon

log='/tmp/my_daemon.log'
#mi PC
p1='/home/colorado/Progs/Daemon_weather/'
#otras PC, defaut
p2='/root/Daemon_weather/'

p=$p1

echo "my_daemon by Colorado"

echo "hola deamon" >>$log
echo $1 >>$log
cd $p

case "$1" in
  start)
    echo "Starting script, weather monitor "

    $p/weather_daemon.py start  >>$log
    ;;
  stop)
    echo "Stopping script weather monitor"

    $p/weather_daemon.py stop
    ;;
  *)
    echo "Usage: /etc/init.d/blah {start|stop}"
    exit 1
    ;;
esac
date >>$log
exit 0
```