

NDBC Data Transfer Methodologies/Discussion (HSU CICORE)

Author: Mike Gough
Organization: HSU CICORE
E-mail: gis@mikegough.net

July 2007

Introduction:

The following document describes the methods used to generate and transfer real-time water quality data to the NDBC website from a LINUX/UNIX system. This document also includes a discussion of some the problems encountered during this process, and the steps taken to resolve them. The methods and scripts described in this document are specific to the HSU CICORE server, and would need to be modified for individual systems.

Methodologies:

Step 1: Create two cron job entries which run bash shell scripts approximately every 15 minutes which generate the xml files containing the most recent data from our two real-time stations (Trinidad & dockb) in the format required by the NDBC. The cron job entries, as well as the contents of each script, are given below.

Step 2: Create a bash shell script which transfers the most recent xml files to the ndbc server every minute via FTP. The script is run with the following command line:

```
#> nohup ndbc_ftp_sleep.sh &
```

The contents of the ndbc_ftp_sleep.sh script are given below.

Discussion:

The document entitled "HOW TO SUBMIT METEOROLOGICAL AND OCEANOGRAPHIC DATA TO NOAA'S NATIONAL DATA BUOY CENTER" was very useful in providing the information necessary to put together the scripts which are used to generate the xml data files.

There biggest obstacle in getting this data submission process up and running, was in getting the automated FTP to work. Putting the ftp script on a cronjob was not working properly. Most problems related to scripts called via a cronjob are due to the difference in user environments between the user whom created the script and the default shell environment variables set up by crontab. I still do not know whether or not this was the problem. I thought I had the problem fixed, but it still wasn't working so I tried a different approach: I put the automated FTP in a script which runs in an infinite loop and

is called with the `nohup` command and sent to the background. The **nohup** command is used to ensure that the script continues to run in an infinite loop even after I have logged out (calling the script with the `nohup` command ignores all hangup signals). The “&” at the end of the command sends the process to the background.

Once the FTP script was put in an infinite loop, there were two other changes that needed to be made. I still don't understand why these changes were required in order to allow for the data to be successfully transferred, but they were.

Change 1: “`epsv4`” needed to be added to the FTP script to ensure the successful transmission of data. This enables extended passive mode.

Change 2: I initially had the automated FTP script set to transfer files at approximately 15 minute intervals (after the xml files had been created). For some reason, the files wouldn't “stick” on the ndbc server though. Increasing the transfer time from 15 minutes to 1 minute fixed the problem (still not sure why).

There is a shortcoming to having the FTP script running in an infinite loop instead of being called via a cronjob – namely, if the power shuts off, or if there is a power surge, the script ceases to run. It must then be manually run from the command line.

In putting together the scripts, other things to be aware of include ensuring that your timestamp is in UTC, and ensuring that you are using the correct depth tag. There are a few different depth tags listed in the instructional document. Initially, I was using the `<tide1>` tag, but later learned that this needed to be changed to the `<dp001>` tag.

Scripts:

Step 1, Cron job entries:

```
01,16,30,45 * * * * /srv/www/htdocs/cicore/script/realtime/ndbc_dockb.sh >
/srv/www/htdocs/cicore/data/real_time/ndbc/ndbc_dockb.xml
```

```
01,16,30,45 * * * * /srv/www/htdocs/cicore/script/realtime/ndbc_trinidad.sh >
/srv/www/htdocs/cicore/data/real_time/ndbc/ndbc_trinidad.xml
```

Step 1, Script 1: `ndbc_dockb.sh`

```
#!/etc/profile-allpaths
echo '<?xml version="1.0" encoding="ISO-8859-1"?>'
echo -e "\t <message>"
echo -e "\t <station>ERKC1</station>"
```

```
echo -e "\t <date>\`usr/bin/psql -d cicore -t -U username -c "select to_char(time + interval '7 hours', 'mm/dd/yyyy HH24:MI') from dockb where time=(select max(time) from dockb)"` </date>" | sed 's/> />/g' | sed 's/ </</g'
```

```
echo -e "\t <met>"  
echo -e "\t<fm64iii>830</fm64iii>"  
echo -e "\t<fm64xx>99</fm64xx>"  
echo -e "\t<fm64k1>7</fm64k1>"  
echo -e "\t<fm64k2>1</fm64k2>"  
echo -e "\t<tp001>\`usr/bin/psql -d cicore -t -U username -c "select temp from dockb where time=(select max(time) from dockb)"` </tp001>" | sed 's/ //g'
```

```
echo -e "\t<sp001>\`usr/bin/psql -d cicore -t -U username -c "select salinity from dockb where time=(select max(time) from dockb)"` </sp001>" | sed 's/ //g'
```

```
echo -e "\t<zox3>\`usr/bin/psql -d cicore -t -U username -c "select disoxy from dockb where time=(select max(time) from dockb)"` </zox3>" | sed 's/ //g'
```

```
echo -e "\t<zox4>\`usr/bin/psql -d cicore -t -U username -c "select do_conc from dockb where time=(select max(time) from dockb)"` </zox4>" | sed 's/ //g'
```

```
echo -e "\t<ztrb1>\`usr/bin/psql -d cicore -t -U username -c "select turbidity from dockb where time=(select max(time) from dockb)"` </ztrb1>" | sed 's/ //g'
```

```
echo -e "\t<zchl1>\`usr/bin/psql -d cicore -t -U username -c "select chlorophyll from dockb where time=(select max(time) from dockb)"` </zchl1>" | sed 's/ //g'
```

```
echo -e "\t<dp001>\`usr/bin/psql -d cicore -t -U username -c "select depth * .3048 from dockb where time=(select max(time) from dockb)"` </dp001>" | sed 's/ //g'  
echo -e "\t </met>"  
echo -e "\t </message>"
```

Step 1, Script 2: ndbc_trinidad.sh

```
#. /etc/profile-allpaths  
echo '<?xml version="1.0" encoding="ISO-8859-1"?>'  
echo -e "\t <message>"  
echo -e "\t <station>TDPC1</station>"
```

```
echo -e "\t <date>\`usr/bin/psql -d cicore -t -U username -c "select to_char(time + interval '7 hours', 'mm/dd/yyyy HH24:MI') from trinidad where time=(select max(time) from trinidad)"` </date>" | sed 's/> />/g' | sed 's/ </</g'
```

```
echo -e "\t <met>"  
echo -e "\t<fm64iii>830</fm64iii>"  
echo -e "\t<fm64xx>99</fm64xx>"
```

```

echo -e "\t\t<fm64k1>7</fm64k1>"
echo -e "\t\t<fm64k2>1</fm64k2>"
echo -e "\t\t<tp001>/usr/bin/psql -d cicore -t -U username -c "select temp from trinidad
where time=(select max(time) from trinidad)"` </tp001>" | sed 's/ //g'

echo -e "\t\t<sp001>/usr/bin/psql -d cicore -t -U username -c "select salinity from
trinidad where time=(select max(time) from trinidad)"` </sp001>" | sed 's/ //g'

echo -e "\t\t<zox3>/usr/bin/psql -d cicore -t -U username -c "select disoxy from
trinidad where time=(select max(time) from trinidad)"` </zox3>" | sed 's/ //g'

echo -e "\t\t<zox4>/usr/bin/psql -d cicore -t -U username -c "select do_conc from
trinidad where time=(select max(time) from trinidad)"` </zox4>" | sed 's/ //g'

echo -e "\t\t<ztrb1>/usr/bin/psql -d cicore -t -U username -c "select turbidity from
trinidad where time=(select max(time) from trinidad)"` </ztrb1>" | sed 's/ //g'

echo -e "\t\t<zchl1>/usr/bin/psql -d cicore -t -U username -c "select chlorophyll from
trinidad where time=(select max(time) from trinidad)"` </zchl1>" | sed 's/ //g'

echo -e "\t\t<dp001>/usr/bin/psql -d cicore -t -U username -c "select depth * .3048
from trinidad where time=(select max(time) from trinidad)"` </dp001>" | sed 's/ //g'

echo -e "\t </met>"
echo -e "\t </message>"

```

Step 2, Script 1: ndbc_ftp_sleep.sh

```

#!/bin/sh

while [ 1 ] ; do

ftp -n -i -v comms.ndbc.noaa.gov <<SCRIPT
user sfbeams S0urd0ugh
epsv4
put ndbc_dockb.xml
put ndbc_trinidad.xml
quit
SCRIPT

sleep 60

rm nohup.out
done

```