# Space Details

| | |
|---|---|
| **Key:** | O3S |
| **Name:** | Oasis Support Engineering |
| **Description:** | |
| **Creator (Creation Date):** | rich (Oct 03, 2007) |
| **Last Modifier (Mod. Date):** | kgomes (Feb 03, 2012) |

## Available Pages

- OASIS Support Engineering Wiki
  - 2008M2 *Deployed on 11-Apr-2008
  - 2011M2H
  - Deployments
  - Iridium1
  - M1 February 2012
  - M1 November 2007
  - M1,M2 and OA Downloads
  - Mooring Processing
  - OASIS Mooring turn
  - Support Engineering Upgrades
    - February 3, 2012
  - Tasks
- Bogus yyyyddd error files....

# OASIS Support Engineering Wiki

This page last changed on Feb 03, 2012 by kgomes.

Welcome to the support pages for the OASIS Mooring Systems. This confluence space documents the various aspects of the OASIS controllers and the moorings that are run by them. This space contains many technical documents related to the OASIS systems as well as operation procedures and logs as well.

## Design Documents

1. MBARI OASIS3 Mooring Guide
2. OASIS To SSDS Documentation

## Operational Documents

1. OASIS Hourly Downloads
2. Mooring Processing
3. OASIS Mooring turn

## Development

1. JIRA Tasks
2. Support Engineering Upgrades

## Other Documents

1. Developer Documentation
    a. Other

1. a. Note attachment on this page for OASIS_TURNS.doc

## Deployment Logs

1. M1 Mooring
    a. November 2007
    b. October 2011

1. M2 Mooring
    a. April 2008

1. M46044 Mooring
    a. 2011M2H

## 2008M2 *Deployed on 11-Apr-2008

This page last changed on May 02, 2008 by rich.

1. roadmap
2. oasis.cfg
3. SSDS_instruments.xls

# 2011M2H

This page last changed on Jul 07, 2011 by kgomes.

Some notes from the deployment of the new Hybrid buoy (M46044):

Originally, the old M2 was going to go silent and then be recovered, but that slipped a few months, so Francisco wanted the old M2 turned back on.

## Re-enabling old M2

#### SSDS enabling

After Rich did all the changes to enable the M2 downloads and then moved the M2H over the M46044, I made the following changes to enable the SSDS processing:

1. Created XML for 1379 and 1796
2. Changed the ssds.cfg on the M

## Deployments

This page last changed on Oct 03, 2007 by rich.

2007M1

# Iridium1

This page last changed on May 18, 2009 by rich.

iridium beacons on M1 and M2

iridium contract/subscription is managed by?

iridium1 acct on tsunami was set up by Pat Allen to receive emails from beacons

cronjob run by iridium1 execs shell script to process data (emails)  into tsunami//oasis/iridum1 directories

```
[8] % whoami
iridium1
[9] % crontab -l
0,5,10,15,20,25,30,35,40,45,50,55 * * * * /oasis/iridium/processiridium

[10] %
```

Note: Im not sure processiridium is working correctly - looks like it should be pruning the text0nnnnn files -rs

here is the script

```
 if [ -s /var/mail/iridium1 ]
then
/bin/cp /var/mail/iridium1 /oasis/iridium/iridium1/incoming

/usr/local/bin/ripmime -i /oasis/iridium/iridium1/incoming -d /oasis/iridium/iri
dium1
rm /oasis/iridium/iridium1/textfile0
DT=`/bin/date +%y%m%d`
if [ -f /oasis/iridium/iridium1/*.sbd ]
then
cat -r /oasis/iridium/iridium1/*.sbd /oasis/iridium/newline >>/oasis/iridium/iri
dium1/$DT

rm /oasis/iridium/iridium1/*.sbd
fi

cat  /oasis/iridium/iridium1/textfile* >>/oasis/iridium/iridium1/text$DT
rm /oasis/iridium/iridium1/textfile*
rm /oasis/iridium/iridium1/incoming
/bin/cp /dev/null /var/mail/iridium1
fi
```

# M1 February 2012

This page last changed on Feb 03, 2012 by kgomes.

This document details the notes associated with the M1 Mooring turn for October 2011.

#### Schedule

| Milestone | Date |
|---|---|
| Initial Configuration | 7/7 |
| Final Science Instrument list | 7/15 |
| Instruments Due | 7/29 |
| Oasis Software development | 7/18 - 8/9 |
| Buoy Mechanical modifications | 7/18 - 8/9 |
| Oasis Code Freeze | 8/10 |
| System Testing | 8/11 - 8/24 |
| Sensor installation | 8/25 - 9/20 |
| Dockside Testing | 9/21- 26 |
| Load on Pt Sur | 9/27 |
| Deployment and Recovery | 9/28 |

#### Sensor Listing

Here is a Word Doc that details the sensors to be deployed on the M1 Mooring in October. I converted that to a table here:

| SENSOR | SERIAL NO | SSDS | LOCATION |
|---|---|---|---|
| Oasis-3 | | | tower |
| E-meter | 041342 | 1602 | tower |
| Grnd fault | 2 | 1403 | tower |
| GPS | 80721087 | 1468 | tower |
| Radio (Viper?) | | | tower |
| Metsys | 1 | 1397 | tower |
| Windbird | 8 | | tower |
| Sonic | A224007 | | tower |
| AT/RH | 93250 | | tower |
| AsimetLWR | 305 | 1251 | tower |
| AsimetSWR | 338 | 1250 | tower |

| | | | |
|---|---|---|---|
| HyperOCR | 229 | 1450 | tower |
| WETStar | 236 | 1425 | tower |
| pCO2 | IRG2-154 | 1743 | tower |
| EK-60 | | | tower |
| ISUS | N/A | 1678 | elevator |
| MicroCAT | 3424 | 1358 | elevator |
| ADCP | 1384 | 1352 | elevator |
| ADCPpower | 010 | | elevator |
| HydroScat | H2010336 | 1390 | elevator |
| HSshutter | SS031014 | | elevator |
| Pump (fluor) | 52578 | | elevator |
| Optode | 131 | 1552 | elevator |
| pH (self-contained) | | | elevator |
| MicroCAT | 3531 | 1314 | 10m cage |
| HyperOCR-Ed | 228 | 1451 | 10m cage |
| Bioshutter | 063 | | 10m cage |
| HyperOCR-Lu | 203 | 1452 | 10m cage |
| Bioshutter | 064 | | 10m cage |
| EK-60 transducer | | | 10m cage |
| MicroCAT | 3777 | 1356 | 20m cage |
| HyperOCR-Ed | 295 | 1625 | 20m cage |
| Bioshutter | 094 | | 20m cage |
| HyperOCR-Lu | 254 | 1627 | 20m cage |
| Bioshutter | 096 | | 20m cage |
| HydroScat-2 | H2061255 | 1629 | 215m nilspin |
| UIM | 0067 | 1489 | 215m nilspin |
| ECO-FLNTUSB | 237 | 1470 | 215m nilspin |
| Deep O2 | | | 215m nilspin |
| SBE-37-IM at 40 | | | |
| SBE-37-IM at 60p | | | |
| SBE-37-IM at 80 | | | |
| SBE-37-IM at 100p | | | |

| | | | |
|---|---|---|---|
| SBE-37-IM at 150 | | | |
| SBE-37-IM at 200p | | | |
| SBE-37-IM at 250 | | | |
| SBE-37-IM at 300p | | | |

## Comments

Oasis cfg file and app are in CVS at
http://moonjelly.shore.mbari.org/cgi-bin/viewvc.cgi/oasis3/deployments/2012M1/

Posted by rich at Apr 16, 2012.

## M1 November 2007

This page last changed on Feb 03, 2012 by kgomes.

1. roadmap
2. oasis.cfg
3. SSDS_instruments.xls

# M1,M2 and OA Downloads

This page last changed on Jul 21, 2014 by rich.

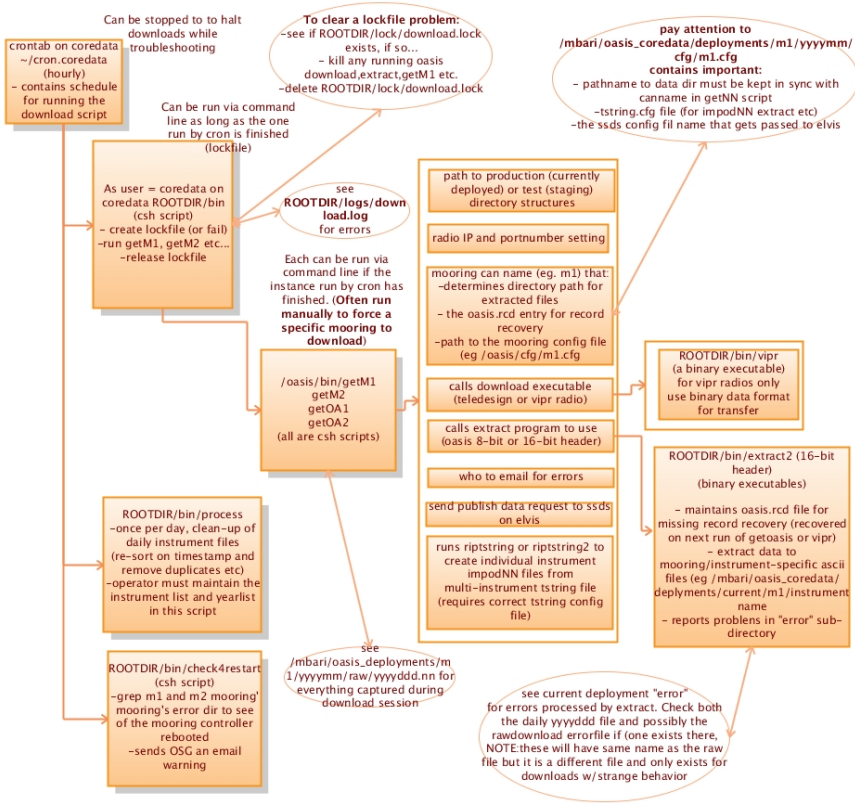
Oasis downloads an are run via cron on workstation **coredata as user 'coredata'.**

**NOTE: in all text and diagrams "ROOTDIR" = /u/coredata/moorings/downloads**

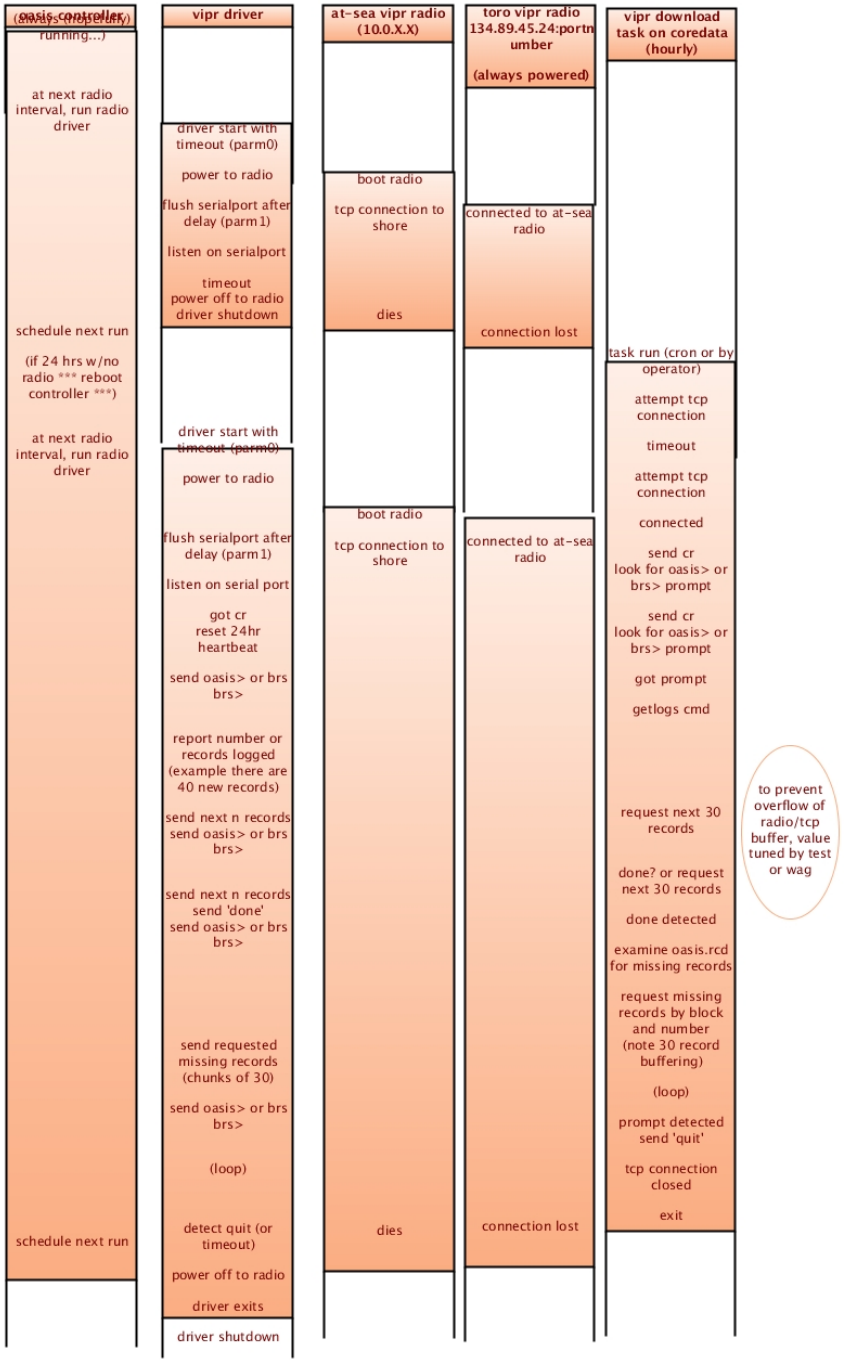
- cd to ROOTDIR
- Scripts and executables are in ROOTDIR/bin
- Downloaded data go to /mbari/oasis_coredata/deployments/m1/yyyymm/raw (where yyymm=year and month of start of deployment)
- Extracted data ascii instrument data files go to mooring-specific dirs at /mbari/oasis_coredata/deployments/m1/yyyymm/m1 and are ingested into ssds (m1 only)

Downloads occur typically hourly at the top of the hour. See ~/cron.coredata OR do a crontab -l to see current schedule

**NOTE: in all text and diagrams "ROOTDIR" = /u/coredata/moorings/downloads**



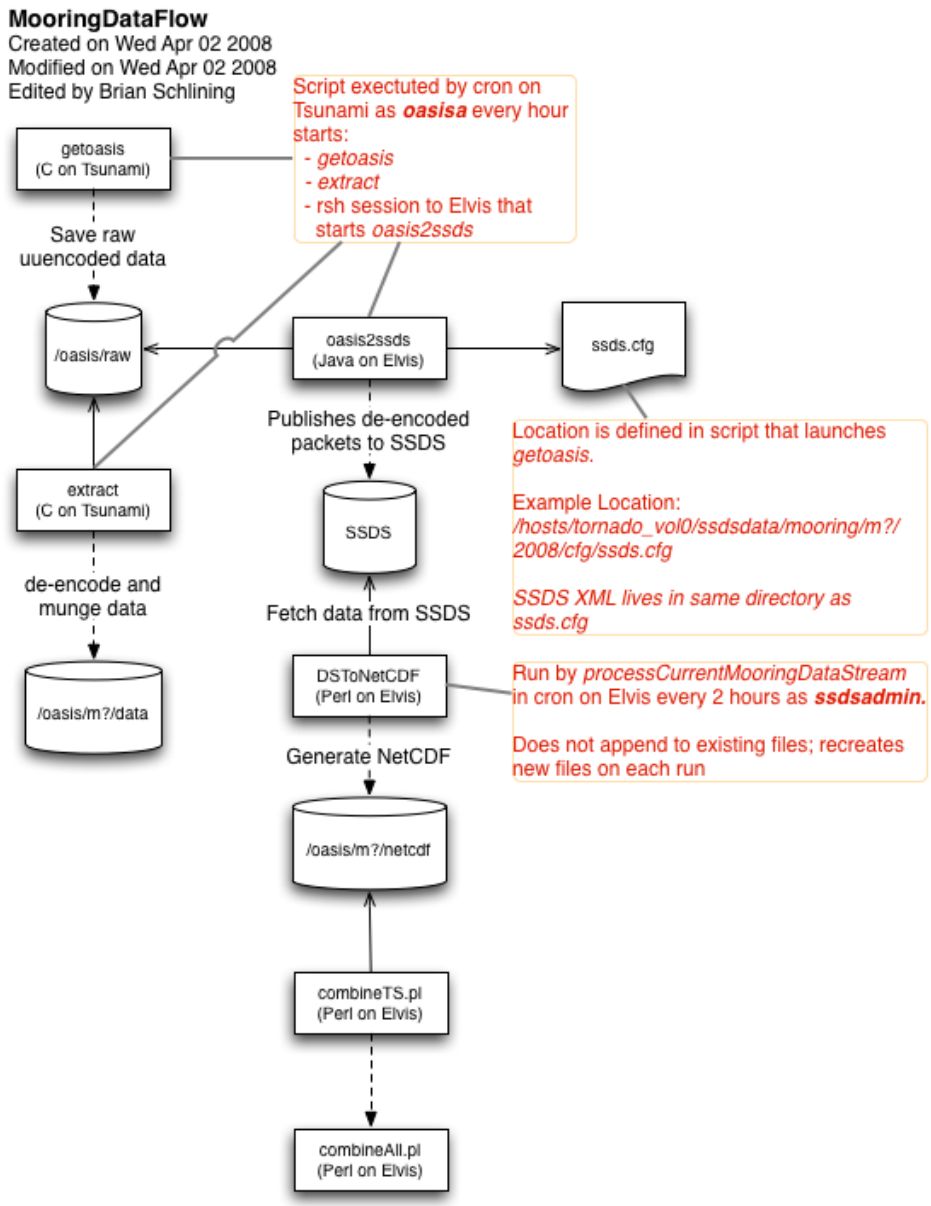

Approx timeline for downloads via vipr radio is shown below.


oasis controller (always running...)
vipr driver
at-sea vipr radio (10.0.X.X)
toro vipr radio 134.89.45.24:portnumber
(always powered)
vipr download task on coredata (hourly)
at next radio interval, run radio driver
driver start with timeout (parm0)
power to radio
flush serialport after delay (parm1)
listen on serialport
timeout
power off to radio
driver shutdown
boot radio
tcp connection to shore
dies
connected to at-sea radio
connection lost
schedule next run
(if 24 hrs w/no radio *** reboot controller ***)
task run (cron or by operator)
attempt tcp connection
timeout
attempt tcp connection
connected
send cr
look for oasis> or brs> prompt
send cr
look for oasis> or brs> prompt
got prompt
getlogs cmd
at next radio interval, run radio driver
driver start with timeout (parm0)
power to radio
flush serialport after delay (parm1)
listen on serial port
got cr
reset 24hr heartbeat
send oasis> or brs brs>
report number or records logged (example there are 40 new records)
send next n records
send oasis> or brs brs>
request next 30 records
done? or request next 30 records
done detected
examine oasis.rcd for missing records
request missing records by block and number (note 30 record buffering)
(loop)
prompt detected send 'quit'
tcp connection closed
exit
to prevent overflow of radio/tcp buffer, value tuned by test or wag
send next n records
send 'done'
send oasis> or brs brs>
send requested missing records (chunks of 30)
send oasis> or brs brs>
(loop)
detect quit (or timeout)
power off to radio
driver exits
driver shutdown
schedule next run

# Mooring Processing

This page last changed on Feb 03, 2012 by kgomes.

## Mooring Processing overview



## OASIS Mooring turn

This page last changed on Jun 14, 2021 by kgomes.





 WARNING: This page is now located in the SE-IE Markdown Git project located here. You can check out that project, edit the documentation and push back to BitBucket and it will automatically be deployed on MBARI's documenation site



This is the procedure to take when the Observatory Support Group turns an OASIS mooring.  A turn means that the currently deployed mooring is recovered and a new one is deployed with a different set of instruments at the same nominal location.

To ease the stress of SSDS data processing continuity on the day of the turn all the instrument XML and metadata processing can be tested with the dockside test deployment that is already part of sta/ ndard OSG operating procedure. This test deployment will get added to SSDS, clearly labeled as a test deployment and the netCDF files and plots will be generated for complete end-to-end testing of the data processing path before the actual turn. When the actual deployment (the turn) happens we just need to change the name of the top-level deployment, e.g. from "Test M1 - November 2007" to "M1 - November 2007".

# A. Procedure for setting up a test mooring deployment

This procedure coincides with standard operation procedure for assembling instruments on a mooring that is about to be deployed. As soon as the mooring configuration spreadsheet is distributed, the OASIS can is closed, and data begins flowing via the getM1 & getM2 scripts on tsunami these steps can be followed to set up a set of test deployments and corresponding netCDF files and plots using the DStoNetCDF.pl and combineTS.pl, combineAll.pl scripts in the DPforSSDS/cimt project. This test can help validate instrument metadata xml and the ssds.cfg file as well as produce a complete end-to-end test of the full SSDS data path.

1. Copy SSDS instrument metadata config files from previous deployment to new deployment to create a starting place (these steps are best performed logged in as ssdsadmin on elvis), e.g.:

   ```
   cd /mbari/ssdsdata/mooring/m1
   cp -r 2006 2007
   ```

   (Of course you will copy the previous year's deployment configuration to a directory for the current year, hereafter referred to as <YYYY> or *yyyy*.) The new directory will have subdirectories named data, cfg, and xml. OSG will email a spreadsheet for the new deplolyment configuration; you may save it in the *yyyy* directory.
2. Edit the mooring .cfg file and change the instrument deviceIDs and the path to the xml files to the newly deployed deviceIDs (aka ISI_IDs) and new xml directory. Note that the deviceID is repeated on each line: once in a field by itself and again in the name of the XML file describing the device deployment:

   ```
   cd <YYYY>/cfg
   vi ssds.cfg
   ```

   Refer to Bob's documentation for details on the formatting of the .cfg file. Generally, this is a tedious editing process where the deviceIDs from the OSG spreadsheet are copied into the proper field and into the name of the xml file that contains the device metadata.
3. Get the most recent XML file for each instrument ID in the ssds.cfg file from the CVS *puckxml* module. Make sure that no deployment specific information is in the .xml files except for

nominalDepth where appropriate. Another exception is the parent platform deployment XML file; its Deployment element should contain name, startDate, nominalLatitude and nominalLongitude attributes so that the parent deployment may be found using the Explorer application, e.g.:

```
<Deployment role="platform" name="Test M1 - September 2008" startDate="2008-09-05T18:00:00Z" \
nominalLatitude="36.764" nominalLongitude="-122.046">
```

In the XML make sure RecordVariable names are not set to standard coordinate axis names (longitude, latitude, depth, time), these are reserved for the OceanSITES data sets which derive from the insturment netCDF files produced with this metadata. Instead choose specific names, e.g. 'MetsysTime' for the Metsys time field. After cleaning up the XML check it back into CVS. Using the OASIS-specific schema, http://dods.mbari.org/data/ssdsdata/config/schema/2004oasis/SSDS_Metadata.xsd, will help with editing using a tool such as Oxygen or JEdit. Copy the files into the *yyyy*/xml subdirectory. Be careful with Device attributes, generally if a Device exists in the SSDS Device table with accurate attributes only the id needs to be specified in the xml file; **if you include other attributes they will overwrite what is in the database.**

4. Special treatment for the TString "instrument": The inductive modem microcats are configured as child deployments of the Inductive modem Comm Device with one output that contains all the microcat data. Examine this xml file with a fine-tooth comb and make sure that the pressure sensor Record Variables are correct as well as the parseRegEx and nominalDepth attributes. [Note: With the M1 - October 2009 deployment we added the individual inductive modem microcats as children of the mooring and also configured a parallel deployment of the TString that produced the same data. With the M2 - April 2010 deployment we configured the individual IM mirocrocats and did not configure a parallel TString deployment. This is a better model for the system and is much easier to configure and produces more easily consumed data by downstream processes such as combineTS.pl.] With future deployments we will not configure TString.
5. Make sure that the m1.cfg file in the cfg directory on oasis has a line referring to the correct ssds.cfg file, e.g.:

```
# The following is for ingesting OASIS data into SSDS
# Please leave it alone; 'extract' et al will ignore it
# 30 Oct 2007 Mike McCann
ssds    /ssdsdata/mooring/m1/2007/cfg/ssds.cfg
```

6. Make sure that the ___ProcessRun lines in ssds.cfg refer to actual files in the xml directory.
7. Check the ___ProcessRun.xml files for correct DataFile and Resource uri/urls.
8. Make sure that the getM? script on tsunami executes the oasisToSSDS program on elvis. The script should have lines like these:

```
set SEND_SSDS $TRUE

# Send the data to SSDS
# Added 20 Jan 2005 Bob Herlien
#
if ( $SEND_SSDS == $TRUE ) then
echo "remsh elvis -l ssdsadmin dev/DPforSSDS/oasis/bin/oasisToSSDS -c $CFGFILE $RAW/$DATAFILE"
remsh elvis -l ssdsadmin dev/DPforSSDS/oasis/bin/oasisToSSDS -c $CFGFILE $RAW/$DATAFILE &
endif
```

9. Before initiating the oasisToSSDS program take a printout of the ssds.cfg file over to the Mooring and verify the Device IDs for each instrument attached to the mooring. It's important to have this important metadata correct from the beginning otherwise the DataStreams will end up in the wrong "buckets" requiring some detailed re-work after the fact, namely following the instrument swap procedure and doing a deep delete (e.g.: http://new-ssds.mbari.org:8080/servlet/MetadataAccessServlet?responseType=text&delimiter=\|&objectToInvokeOn=DataProducerAccess&method=deepDelete&p1Type=DataProducer&p1Value=DataProducer|id=<deploymentID>) on the wrong instrument deployment.
10. Test by:
    a. Monitor the extractRawData.log file in the ssdsdata/mooring/logs file for progress and any INFO or ERROR messages. For instance if you see something like "[SSDS:main] INFO oasis.ssds.ingest.OasisToSSDS - Unprocessed record, type = ASIMET_HRH" then that is an instrument missing from the ssds.cfg file for which data records are being received. To fix this add the correct instrument record to ssds.cfg.
    b. Check for metadata ingest in Explorer by searching for 'Test' in the "Search By Deployment Name:" field (note: may need to hit 'Enter' a few times to actually get something returned). You should see a tree structure headed by the Deployment name entered in step 3. Open the deployment and drill down through the Attached Devices and Outputs to spot check the mooring instruments and their attributes - confirm against the the list provided by OSG. If

there are any problems check the insturment XML and the ssds.cfg files configured in steps 2 to 4.

c. To correct missing or wrong Deplyment attributes (e.g. nominalLatitude and nominalLongitude) the quickest way to correct is to use Enterprise Manager to select the DataProducer (identified by id in Explorer) and edt the fields directly in the table.

11. Set up netCDF creation and plotting:
    a. Examine the processCurrentMooringDataStreams.sh script in /u/ssdsadmin/dev/DPforSSDS/cimt for lines (probably commented out) that process the test deployment. They will look something like:

```
#
# Test M1 - Combine scripts look in Deployment Directory m1/200710
#
##set DDIR=200710
##DStoNetCDF.pl -mooring M1Test -ssdsServer new-ssds.mbari.org -ssdsDataServer new-ssds.mbari.org \
-outputDir $ODIR -verbose -current |& tee $ODIR/m1/processing.log
```

    set DDIR to the YYYYMM of the test deployment and execute the DStoNetCDF.pl script with the above arguments; ODIR = /mbari/ssdsdata/deployments on elvis. Note that the name of the Deployment is "Test M1 - <startDate>" in SSDS, but that the argument to the -mooring parameter is "M1Test". The DStoNetCDF.pl script contains a lookup that matches "M1Test" to "Test M1 ...".

    b. Execution of DstoNetCDF.pl will produce a web page in ssdsdata/deployments/m1test/current_qcPlots.html that contains links to the produced netCDF files and plots (available as web page http://dods.mbari.org/data/ssdsdata/deployments/m1test/current_qcPlots.html). Examine this page to see that every instrument we can process has a netCDF file and corresponding plots. If there are problems follow the instructions in the TROUBLESHOOT file in /u/ssdsadmin/dev/DPforSSDS/cimt. Generally, if the metadata is correct in SSDS then the netCDFs and plots will be produced. Either update the .cfg and XML files following steps 2-4 or update SSDS directly as appropriate.

    c. Configure the DStoNetCDF.pl to run for the test deployment in the cron job and inform OSG and the OASIS discussion list of the availability of these products.

# B. Procedure for doing the actual production mooring turn

### Close existing mooring deployment

1. So, after the test has been run and data validated there is a sequence of manual steps for closing the previous deployment and beginning the production data processing of the new deployment. On the day the previous production mooring deployment is ended close that deployment by setting the endDate for the parent platform and child deployments. From SSDS Explorer find the DataProducer ID for the mooring deployment, do a SELECT for that record and edit the endDate field (the times in the database are GMT). Then select all child deployments with a query on the foreign key like this:

```
SELECT     *
FROM         ssdsdba.DataProducer
WHERE     (ParentID_FK = 27122)
```

(Make sure to use the DataProducer ID for the platform deployment.) Set all endDates that are <NULL> to the actual end date.

For instruments that have child Sensor deployments (e.g. the Hyperspectral radiometers and imctd) do the same thing by changing the ParentID_FK to the id of the instrument and set all the <NULL> endDates so that everything on the recovered mooring is closed (or...as below -rschramm 4/2010)

```
declare @myID as bigint
set @myID = 31548
SELECT id,name,dataProducerType,startDate,endDate,ParentID_FK
FROM [ssdsdba].[DataProducer]
WHERE id = @myID and endDate is NULL
UNION
SELECT id,name,dataProducerType,startDate,endDate,ParentID_FK
FROM [ssdsdba].[DataProducer]
WHERE parentid_fk = @myID  and endDate is NULL
```

```
UNION
SELECT id,name,dataProducerType,startDate,endDate,ParentID_FK
FROM [ssdsdba].[DataProducer]
WHERE parentid_fk in ( SELECT id
FROM [ssdsdba].[DataProducer]
WHERE parentid_fk = @myID  and endDate is NULL)
ORDER BY id
```

**Configure new mooring deployment**

1. To control the new SSDS Metadata ingest temporarily turn off the oasisToSSDS execution in the getM? script on tsunami. This way you may edit the xml files at leisure without the ingest picking up any incorrect metadata while you are in the process of editing. In the XML file for the new platform deployment, which is currently configured as the Test deployment, edit the name, startDate, and nominalLatitude and nominalLongitude attributes to reflect the production deployment. E.g.:

   ```
    <Deployment role="platform" name="M1 - October 2008" startDate="2008-10-08T17:00:00Z"
   nominalLatitude="36.764" nominalLongitude="-122.046">
   <Device id="1306"/>
   <Resource
   url="doc://foobar/watchCircle?centerLon=-122.0323&amp;centerLat=36.7562&amp;warningDist=1.2"
   name="Watch circle information">
   <description>Data for this mooring's watch circle are embedded in the uriString of this
   Resource. An application may parse for these parameters to use as criteria for
   issuing a warning if the GPS position is geater than warningDist (in km) from
   centerLat and centerLon (in decimal degrees WGS84)</description>
   </Resource>
   </Deployment>
   ```

   The nominalLatitude and nominalLongitude values should be exactly the same as all other deployments at M1 or M2. Make sure that the watchCircle parameters are relatively correct, the centerLon and centerLat values may be changed to reflect the actual anchor location. And these values can be updated as data come and in and we get a better idea of the actual watch circle. Save the changes to this file, check those changes into the puckxml CVS project and touch the remaining xml files so that SSDS ingest will recognize them as new.
2. Do not close (set endDates) the existing Test deployment until after the oasisToSSDS has run with the new platform deployment name and all new instrument deployments have been created in SSDS_Metadata.
3. See that oasisToSSDS is allowed to execute in the getM? script.  Monitor the /mbari/ssdsdata/mooring/logs/extractRawData.log file to see that downloaded records for the mooring being turned are processed. Then check that metadata is properly loaded with a query looking at the recently ingested Deployment metadata, e.g.:

   ```
   SELECT     id, name, startDate, endDate, ParentID_FK
   FROM       ssdsdba.DataProducer
   WHERE      (dataProducerType = 'Deployment')
   ORDER BY id DESC
   ```

   Things to check for:

   a. All ParentIDs should not be null except for the Platform deployment.
   b. Sensors should be children of instruments, e.g. inductive modem microcats are children of the Surface Inductive Modem
   c. All other instruments should be children of the platform deployment
4. Check the SSDS_Metadata database for successful ingest of the platform and instrument deployment metadata with a query like this:

   ```
   SELECT     ssdsdba.DataProducer.id, ssdsdba.DataProducer.ParentID_FK AS ParentID,
   ssdsdba.DataProducer.name,
   ssdsdba.Device.id AS DeviceID,
   ssdsdba.Device.name AS DeviceName
   FROM         ssdsdba.DataProducer INNER JOIN
   ssdsdba.Device ON ssdsdba.DataProducer.DeviceID_FK = ssdsdba.Device.id
   WHERE     (ssdsdba.DataProducer.id  >= 28873)
   ORDER BY ssdsdba.DataProducer.id DESC
   ```

You will need to set the DataProducer.id criteria to be appropriate for this deployment.

5. Add the new deployment to the DataProducerGroup "Mooring Deployments". Do this query:

```
SELECT     ssdsdba.DataProducerAssocDataProducerGroup.*
FROM         ssdsdba.DataProducerAssocDataProducerGroup
WHERE     (DataProducerGroupID_FK = 111)
```

   and add a record to the DataProducerAssocDataProducerGroup table for the new platform deployment and the group id 111, which is the MooringDeployments group.
6. You next need to link up the new deployment to a DataProducerGroup that contains all the deployment of the like designator. For example, if an M1 turn is being done, you need to add the new deployment to the "M1 Deployments" DataProducerGroup. You first need to find the DataProducerGroup.id for the group you are looking for. Do this query in Enterprise Manager:

```
SELECT * FROM ssdsdba.DataProducerGroup WHERE name = 'M1 Deployments'
```

   Make sure you use the correct *M* Number for the turn you are doing. Note the id from the results. Now you need to add the new deployment to that data producer group. Right-click on the DataProducerAssocDataProducerGroup table and do 'Open Table->Return All Rows'. You can then scroll to the bottom and there should be an open row that you can insert new values in. Put the ID of the new deployment in the first column and the id from the previous query in the second and then click out of the new row (that should commit the row). Do the same for the 'Mooring Deployments' DataProducerGroup.
7. If a device is missing as a child of the mooring deployment then use the Raw Data Access Page to query for metadata from the device. For instance if the GPS is not a a child (as can now be seen from using SSDS Explorer and the predefined 'Mooring Deployments' query) then enter its device id and '0' for record type. From Enterprise manager you can then see which deployment and parent it got assigned to, e.g.:

```
SELECT     ssdsdba.DataProducer.id, ssdsdba.DataProducer.ParentID_FK AS ParentID,
ssdsdba.DataProducer.name,
ssdsdba.Device.id AS DeviceID,
ssdsdba.Device.name AS DeviceName
FROM         ssdsdba.DataProducer INNER JOIN
ssdsdba.Device ON ssdsdba.DataProducer.DeviceID_FK = ssdsdba.Device.id
WHERE     (ssdsdba.Device.id = 1468)
ORDER BY ssdsdba.DataProducer.id DESC
```

   If there is no deployment for the time of the XML ingest then there was probably a deadlock on SSDS's attempt to insert it into the database. You can manually force an insert by altering the text of the device's xml file in moorings/m?/yyyy/xml (just add a space or something) and then run oasisToSSDS again, e.g.: > dev/DPforSSDS/oasis/bin/oasisToSSDS -c /oasis/cfg/m1.cfg /oasis/raw/m1.2007310.02, or wait for it to run with the hourly download on tsunami.
8. Set up netCDF creation and plotting: In the processCurrentMooringDataStreams.sh script change the DDIR environment variable to the current YYYYMM for the production mooring run. Comment out the Test deployment DStoNetCDF.pl execution and add lines for the closed deployment to the DEPLOYMENTS file.
9. You will also need to edit the DStoNetCDF.pl script to set the new name of the M1 deployment, e.g.:

```
#
# SSDS mooring name Deployment lookup
#
my %ssdsMooringDeplNames = (
M0 => 'M0',
M1 => 'M1 - October 2009',
M1Test => 'Test M1 - October 2009',
M2 => 'M2 - April 2009',
```

10. Monitor the SSDS processing web page (e.g. http://dods.mbari.org/data/ssdsdata/deployments/m1/current_qcPlots.html) for successful data processing.
11. When the mooring log message is sent to oasis with the actual time of deployment enter that as the startDate in the database. All times (DTGs) in the SSDS_Metadata database are GMT. Make sure to set the startDate for the imctd microcat sensors too. This can be done with a query like this where you use the proper dataProduceIDs for the mooring and the imcd:

```
SELECT     id, name, ParentID_FK, startDate, endDate, nominalDepth
FROM         ssdsdba.DataProducer
```

```
WHERE     (dataProducerType = 'Deployment') AND (ParentID_FK = 31564) OR
(ParentID_FK = 31548)
ORDER BY id DESC
```

12. Because of some bug in SSDS ingest the dataContainerTypes of the outputs from the instrument deployments do not get properly assigned the values of 'Stream'. This needs to be fixed so that the NDBC datatransfers will work. To fix it edit the SSDS_Metadata database starting with a query like this:

    ```
    SELECT     ssdsdba.DataContainer.*
    FROM         ssdsdba.DataProducer INNER JOIN
    ssdsdba.DataContainer ON ssdsdba.DataProducer.id = ssdsdba.DataContainer.DataProducerID_FK
    WHERE     (ssdsdba.DataProducer.ParentID_FK = 31548)
    ```

    where you use the DataProducerID for the new mooring deployment in the WHERE clause. ==> Change all of the 'File's in the dataContainerType' field to 'Stream's.

# C. Procedures to be done after the mooring turn

**Set up download info deployment and reporting**

Though not a real instrument, we configure a virtual 'dlinfo' instrument for the download scripts to attach download statistics data. We re-use the same device IDs for the M1 and M2 moorings, so it's best to configure this after the new mooring is out and all of those deployments have been closed. As the data are delivered "out of band" from the typical OASIS instruments we need to create a deployment for dlinfo instrument by hand.

1. The most direct way is to use the createDuplicateDeepDeployment service call. For example, to duplicate the 2009 M2 dlinfo deployment for the 2010 M2 deployment this call was executed: http://new-ssds.mbari.org:8080/servlet/MetadataAccessServlet?method=createDuplicateDeepDeployment&objectToInvokeOn=DataProducerAccess&p1Type=moos.ssds.metadata\|id=**34496**&p2Type=Date&p2Value=**2010-04-03T22:00:00Z**&p3Type=boolean&p3Value=false&p4Type=Date&pdownload&p6Type=String&p6Value=&delimiter=|. Of course you will need to adjust the DataProducer ID Date values for the new deployment you are creating. Here is the Key to the parameters:

   ```
   Key:
   ----
   http://localhost:8080/servlet/MetadataAccessServlet
   ?responseType=text
   &delimiter=|
   &objectToInvokeOn=DataProducerAccess
   &method=createDuplicateDeepDeployment
   &p1Type=DataProducer
   &p1Value=DataProducer|id=XXXX (XXXX is the ID of the deployment to copy)
   &p2Type=Date
   &p2Value=XXXXXX (XXXXXX is the start date of the new copy in XML format YYYY-MM-DDTHH:MM:SSZ)
   &p3Type=boolean
   &p3Value=(true|false)  (this is to indicate if you want the original deployment to be closed)
   &p4Type=Date
   &p4Value=XXXXXX (XXXXXX is the end date for the original deployment (if p3Value is true))
   &p5Type=String
   &p5Value=XXXXXX (XXXXXX is the DataProducer name of the new DataProducer)
   &p6Type=String
   &p6Value=XXXXXX (XXXXXX is the base URL to use for the new DataContainers that will be
   created).
   &delimiter=|
   ```

   Executing the createDuplicateDeepDeployment service call will return an ID for the new deployment.

1. Edit the new deployment record in the DataProducer table to adjust it's parentID_FK to be for the new mooring. While there edit times and name as appropriate.
2. Edit the getM? script on tsunami to use the proper device and parent ID (this will be the device ID of the torroid of the mooring deployment) and make sure that the '/oasis/bin/ssdsSubmit.pl $deviceId $parentId "$starttime_es,$endtime_es,$filesize,$rtnsts"' line in the script is configured to run.

Here's another example for the October 2010 M1 turn:

```
http://new-ssds.mbari.org:8080/servlet/MetadataAccessServlet?
method=createDuplicateDeepDeployment&objectToInvokeOn=DataProducerAccess&\
p1Type=moos.ssds.metadata.DataProducer&p1Value=DataProducer|
id=35591&p2Type=Date&p2Value=2010-10-27T21:00:00Z&p3Type=boolean&p3Value=false&p4Type=Date&\
p4Value=2010-q0-27T20:30:00Z&p5Type=String&p5Value=getM1-
download&p6Type=String&p6Value=&delimiter=|

(This returned a new DataProducerID of 41556.  The ParentID_FK for this record in the DataProducer
table was then changed from 35388 to
41197 and the endDate cleared to null.
Also, with the change to alternating deviceIDs implemented in 2010 the deviceID_FK was changed from
1698 to 1768 for this M1 deployment.
```

N.B. A rotating scheme for the virtual device IDs was implemented in 2010. This permits the steps described above to be executed prior to the mooring turn during the extensive dock-side test period.

**Enable processing for other "virtual" devices**

1. Turn on ClockSync processing. We also use a virtual device ID for these data. Simply uncomment the line for it in the ssds.cfg file.
2. Turn on ISUS processing. Simply uncomment the line for it in the ssds.cfg file. (Device ID re-used from previous deployment.)

**Make all of the child instrument deployment start times the same as the mooring start time**

When SSDS receives a packet from an instrument that is not currently deployed it will create a Deployment record (in the DataProducer table) with a startDate that is set to the time of the first record received. As the mooring starts up all of the instrument deployments will have different start times based on when they each first saw data. This can present problems for the data processing that follows, especially for the jobs that aggregate the microcat data into a single ZT file that is used to produce the contour temperature and salinity wind stick plots. To prevent these problems it's best to edit the startDates of the child instrument deployments so that they are all the same. This is currently most easily done through Enterprise Manager with a query like below (this is for the 201010 M1 deployment) to get all the child instrument deployments:

```
SELECT    *
FROM      ssdsdba.DataProducer
WHERE     (ParentID_FK = 41197)
ORDER BY  startDate
```

Then copy and paste the datetime string from one cell to the next. The times are GMT.

**Cycle links to previous deployment**

1. Edit previous.html file in /mbari/ssdsdata/deployments to add a line for the new deployment and add the end date and archive url for the just closed deployment, e.g.:

   ```
   <li><a href="http://dods.mbari.org/data/ssdsdata/deployments/m2/m2_200904_qcPlots.html"
   >200904</a> 2009-04-29 to 2010-04-03</li>
   <li><a href="http://dods.mbari.org/data/ssdsdata/deployments/m2/current_qcPlots.html"
   >201004</a> 2010-04-03 to present</li>
   ```

2. Add lines to DEPLOYMENTS file in dev/DPforSSDS/cimt/ for the just closed deployment, e.g.:

   ```
   DStoNetCDF.pl -mooring M2 -deployment "M2 - April 2009" -ssdsServer new-ssds.mbari.org -
   ssdsDataServer new-ssds.mbari.org -outputDir \
   /mbari/ssdsdata/deployments -procClosed -verbose
   combineM.pl -mooring M2 -ssdsServer new-ssds.mbari.org -deployments 200904 -ssdsDataServer
   new-ssds.mbari.org -inputDir /mbari/ssdsdata/deployments
   combineTS.pl -mooring M2 -ssdsServer new-ssds.mbari.org -deployments 200904 -ssdsDataServer
   new-ssds.mbari.org -inputDir /mbari/ssdsdata/deployments
   combineAll.pl -mooring M2 -ssdsServer new-ssds.mbari.org -deployments 200904 -ssdsDataServer
   new-ssds.mbari.org -inputDir /mbari/ssdsdata/deployments
   ```

and

```
/bin/cp /mbari/ssdsdata/deployments/m2/200904/OS_*.nc /mbari/FTP/pub/OceanSITES
```

3. Execute these lines to generate "closed deployment" data products and web pages and for submission to the OceanSITES GDAC
4. Confirm that the processing executed properly. Sometimes mangled timestamp data weasels its way into the data stream causing instrument netcdf files to be named with bogus start dates. (This can also happen during a deployment and is one of the maintenance tasks one should follow to keep the data flowing to where it needs to go.) The combine__.pl scripts will then create the OceanSITES formatted files encompassing the dates of all the instrument netcdf files. Incorrect instrument file names will create incorrect OS_MBARI* file names causing CenCOOS and NDBC/Ifremer to complain about the correctly named file not being updated with new data. The fix involves a purging of the bad files, double checking the metadata in SSDS and reprocessing, with perhaps additional checks in the instrument processing perl code to skip over bad records. This is best done at the Unix command line by cd'ing to the deployment directory, e.g.

   ```
   cd /mbari/deployments/m1/201010
   ls -lrt
   ```

   and removing files that have bogus dates that do not represent the deployment start date. You should also remove all the directories and files in the gifs/ subdirectory. These will all get recreated when you run DStoNetCDF.pl and the combine__.pl scripts. With luck, simply removing the bad files and rerunning the processing scripts will fix things. If not, identify where the bad dates are coming from and fix as appropriate.

Mike McCann (First edit: 30 October 2007, Last updated: 1 March 2012)

## Comments

On the day of the turn...

the place you change the "top-level deployment name" is on elvis as ssdsadmin. In the /hosts/tornado_vol0/ssdsdata/mooring/m?/200?/xml/nnnn.xml where nnnn.xml is the device id for the oasis buoy itself (the fiberglass doughnut....). Also looks like you set the startDate there as well. Not sure if the startDate can be set later or does it have to be correct from the get-go...

Posted by  at Apr 11, 2008.

Also, Kevin indicates it is necessary to 'touch' all of the xml files in /hosts/tornado_vol0/ssdsdata/mooring/m?/200?/xml/

as part of the turn process to make the new xml ingest.

Posted by  at Apr 11, 2008.

The critical missing piece is documented at:

http://oceana.shore.mbari.org:8082/browse/CD-10

Posted by  at Apr 11, 2008.

# Support Engineering Upgrades

This page last changed on Feb 03, 2012 by kgomes.

In 2012, the Support Engineering Upgrades project set out to clean up, simplify and solidify the OASIS data processing flow to ensure more robust core mooring data.

**Design Meetings**

1. February 3, 2012

# February 3, 2012

This page last changed on Feb 03, 2012 by kgomes.

**Attendees**

1. Rich Schramm
2. Kevin Gomes

**Notes**

This meeting was to start sketching out what needs to be done to simply the OASIS data processing flows.

**Whiteboard images**

1. Board Section 1
2. Board Section 2
3. Board Section 3
4. Board Section 4
5. Board Section 5

**Action Items**

1. Define the routing key topologies for data streams (including monitoring, logging, etc. to setup for DataProbe type consumer).
2. Take the mapping file from OASIS2SSDS and parse so that extract on tsunami side can construct google protocol buffer messages and publish to RabbitMQ
3. Create consumers (or extend MARS consumers) to write message to SSDS tables.
4. Move processing that happens in OASIS2SSDS to consumer so that two data streams can be produced.

## Tasks

This page last changed on Oct 03, 2007 by rich.

### wishlist

## Bogus yyyyddd error files....

This page last changed on Nov 07, 2007 by rich.

```
From: Schramm, Rich rich@mbari.org
Sent: Tuesday, November 06, 2007 3:26 PM
To: Oasis Mooring Discussion List
Subject: oasis occasional bogus oasis timestamp explanation...
Several OSG folks have mentioned annoyance at some of the wacko time-stamped
files that getcreated because of occasional garbled transmissions. The oasis
system is remarkably able tore-request the missing records on subsequent
downloads, however there are occasionally somegenerated garbage files that
remain to be manually cleaned out. I believe I understand where they are
sneaking through the extraction process and what wewould need to do to
fix it should we wish too. This 'fix' would not effect the raw datafiles,
only the generated errors files.

In /oasis/src/oasis3/src/operations/src/extract.c ~ LINE 410
The code snippet below runs in a while-loop and when errors are encountered
it kicks out ofwherever its at using a "continue" to cause the loop to
seek-out the next sync byte in theraw stream... It appears while there
are checks for bad record types etc. The only time checkis for time=0L
(see (klh)08Aug01 below)

My suspicion is that the occasional garbled timestamp passes the unix conversion
to a "valid" gmt but is nonsense to us. The fix would be to reject times "too old"
or into the future by adding a test immediately afterKents 0 test and prior to
computing a new value for the global 'itime'. (we don't want to allowitime to
get set to bogus yyyyddd  because its used to generate filenames...hence the
problem....)"into the future" is easily determined from the cpu clock... what
a reasonable "too old" would need to be decided though we are trying to
clip wild things like more than a couple of years ago... If we don't want
to fix the extract - no biggie as far as Im concerned, we can just tuck this
bit of knowledge away for future reference Thoughts anyone.-Rich:
:
:
:
hdr.log_type = buffer0;
hdr.log_nmbr = getHdrWord(&buffer1, fileType);
hdr.log_len = getHdrWord(&buffer3, fileType);
hdr.log_time = getHdrLong(&buffer5, fileType);tp = gmtime( (time_t *)&hdr.log_time );
/* Error check for bad header (klh) 08aug01*/
if(tp==0L) { print_error("Invalid header log time in %s, record %d (type %d)\n", filename,
hdr.log_nmbr,hdr.log_type);continue; } if ( y2k )
itime = (1000 * (tp->tm_year + 1900)) + tp->tm_yday + 1;
else
itime = (1000 * tp->tm_year) + tp->tm_yday + 1;dtime = tp->tm_yday + 1.0 + (double)((3600 * tp-
>tm_hour)
+ (60 * tp->tm_min) + tp->tm_sec) / 86400.0;len_got = cc - 9; /* compute amount of log data
gotten*/
if ( len_got > 0 )
memmove( buffer, buffer + 9, sizeof(buffer) - 9 );
else if ( len_got < 0 ) /* Move log data to start of buffer*/
{
printf("Incomplete record header in %s, record %d (type %d)\n",
filename, hdr.log_nmbr,hdr.log_type);
continue;
}
:
:
:
:
```