Alaska Earthquake Information Center

University of Alaska Fairbanks

The AEIC Antelope-QDDS Interface

AEIC Internal Report 2007-01

by Glenn Thompson & Mitch Robinson

July 2007

Suggested citation:

Thompson, G. and Robinson, M., 2007. The AEIC Antelope-QDDS Interface, University of Alaska Fairbanks, AEIC Internal Report 2007-01.

This version was last revised: July 2007

This document, any updates to it, and any additional information are available at:
http://www.aeic.alaska.edu/AEIC/internal/report/2007-01/

The Alaska Earthquake Information Center is a cooperative program between the Geophysical Institute of the University of Alaska and the U. S. Geological Survey with support from the Earthquake Hazards Programme.

In order to comply with the USGS's QDDS system, AEIC has found it necessary to develop an interface between Antelope and QDDS. The main purpose of this report is to provide some clues as to how this works.

1. The QDDS system

While the purpose of this report is not to document the QDDS system itself, it makes sense to give some brief information. First of all, limited documentation is at the QDDS ftp site at: ftp://ehzftp.wr.usgs.gov/QDDS/QDDS.html

Stephen Jacobs, a USGS summer employee, developed QDDS in 1998. Further maintenance and development was thereafter done by Alan Jones. The system is distributed as a jar file, containing only bytecodes – i.e. no source code is included.As its written in Java, it is platform independent. The format it uses for encoding earthquake origin information is called CUBE (see link on QDDS ftp site) and its an 80 character string based on punched card limitations. USGS have said for many years they plan to replace QDDS with a system called EIDS which will use a new XML format instead.

The QDDS system at AEIC runs from /home/qdds/run. The listing of this directory is:

```
-rw-r--r--   1 nobody   nobody       241 Dec 16  2003 comm.lst
-rw-r--r--   1 nobody   nobody   2488717 Jul 13 11:49 cronlog
-rw-r--r--   1 nobody   nobody         7 Jul 13 12:13 curr_file_id
drwxrwsr-x   2 nobody   analyst     1024 Jul 13 08:49 logdir/
drwxrwsr-x   2 nobody   analyst   438784 Jul 13 12:13 outputdir/
drwsrwsr-x   2 nobody   analyst     3072 Jul 13 11:32 polldir/
-rw-r--r--   1 nobody   nobody       338 May  7  2002 QDDS.config
-rw-r--r--   1 nobody   nobody       336 May  6  2002 QDDS.config_orig
-rwx------   1 nobody   nobody     55741 Apr 16  2004 QDDS.jar*
-rw-rw-r--   1 nobody   nobody      3921 May  6  2002 QDDS.jar_nohup.log
-rw-rw-r--   1 nobody   nobody       124 Jul 13 12:14 save_max_received
drwxrwsr-x   2 nobody   analyst      512 May  6  2002 storagedir/
drwxrwsr-x   2 nobody   analyst      512 Jul 13 12:13 tempdir/
```

QDDS.jar	The executable code
polldir/	where message files are placed to be broadcast to the hubs.
outputdir/	where message files are put by the hubs, according to the QDDS webpages.
comm.lst	defines to connections to the USGS/IRIS hubs.
logdir/	contains daily log files of qdds activity.
curr_file_id	the message number used for the most recent event message.
storagedir/	empty
tempdir/	empty

The job of the Antelope interface to QDDS then is simply to make sure the right files get placed into polldir/.

1 2. An overview of the Antelope interface to QDDS

No documentation could be found pertaining to the Antelope interface to QDDS, so I have tried to piece it together by following cronjobs and links in programs used by the beeper duty staff. **The path /home/qdds/ is assumed throughout – prepend this to all relative paths given.**

There appear to be four main elements to the interface:

- First, there is a cronjob running orb_qdds which monitors new origin packets on earlybird:6510 and generates a CUBE code with priority 0 for each new event on the orb, and saves this code to run/polldir/orb.$evid and polldirorb/orb.$evid. See section 5.1 for more details.

- Second, there is another cronjob running createqddsdb which generates a CUBE code with priority 5 for each new event in the database /iwrun/op/db/seg/quakes, and saves this code to polldirdb/event.$evid and copies this to run/polldir/event.$evid.

- Third, the beeper duty person manually checks all new events with dbevents that arrive in /Seis/databases/duty/aeic_quakes [1]. From there the beeper duty person can delete a qdds event that should not have been submitted by invoking DELETE_QDDS_EVENTID. In a round-about way what seems to happen is that a modified version of the $aline (CUBE code)with 'E ' changed to 'DE' is written to polldirdelete/del0.$evid, and the original copy of the event file or alarm file is renamed to a capitalised version of itself. polldirdelete/del?.$evid is then also copied to run/polldir/del?.$evid (QDDS system) and renamed to polldirdelete/delete.$evid. See section 6 for more details.

  [1] There may be an issue here. Earlybird saves events to /iwrun/op/db/seg/quakes. Is this the same as /Seis/databases/duty/aeic_quakes? In any case, it might be better to reorganise everything so it all runs off the summary event orb and the summary event database.

- Fourth, the beeper duty person also updates and releases events from dbe if they surpass threshold conditions (after using icetools to call wormwatch and then processing an event with dbloc2). Eventually they will be presented with an 'update qdds' widget, which calls createqddsdb and which in turn calls other programs. Then very roughly (see section 7 for more details):

  1. From /iwrun/op/db/seg/quakes the new event is written to polldirdb/event.$id with a CUBE code priority 5. This is then copied to run/polldir/event.$id (if someone hasn't already deleted the event from QDDS) and moved to polldirfinish/event.$id.

  2. From /home/alarm the new event is written to polldiralarm/event.$id with CUBE code priority 6. This is then copied to run/polldir/alarm.$id (if someone hasn't already deleted the event from QDDS) and moved to polldirfinish/alarm.$id.

  Its by checking if polldirdelete/delete.$id that the Antelope interface to QDDS knows not to broadcast an event or alarm deleted earlier using dbevents.

2 3. Source code

The software is stored at /usr/local/bin/QDDS/. The listing of this directory is:

```
-rwxr-xr-x   1 mitch    staff      3361 Jun 14  2006 createqddsdb*
-rw-rw-r--   1 mitch    staff     13794 Jun 13  2006 dborigin_to_qdds.c
-rw-rw-r--   1 mitch    staff    102600 Jun 28 07:01 dborigin_to_qdds.o
-rwxrwxr-x   1 mitch    staff     36820 Jun 13  2006 db_quake_qdds*
-rw-rw-r--   1 mitch    staff      2337 Jun 13  2006 db_quake_qdds.c
-rw-rw-r--   1 mitch    staff     75412 Jun 28 07:01 db_quake_qdds.o
-rwxrwxr-x   1 mitch    staff      1515 Jun 14  2006 DELETE_QDDS_EVENTID*
-rwxrwxr-x   1 mitch    staff     11924 Jun 13  2006 delete_qdds_eventid*
-rw-rw-r--   1 mitch    staff      2879 Jun 13  2006 delete_qdds_eventid.c
-rw-rw-r--   1 mitch    staff     12412 Jun 13  2006 delete_qdds_eventid.o
-rw-rw-r--   1 mitch    staff       272 Jun 13  2006 Makefile
-rw-rw-r--   1 mitch    staff       202 Jun 13  2006 Makefile_delete_qdds_eventid
-rw-rw-r--   1 mitch    staff       266 Jun 13  2006 Makefile_orb_qdds
-rwxrwxr-x   1 mitch    staff     36816 Jun 13  2006 orb_qdds*
-rw-rw-r--   1 mitch    staff      2682 Jun 13  2006 orb_qdds.c
-rw-rw-r--   1 mitch    staff     72472 Jun 13  2006 orb_qdds.o
-rwxrwxr-x   1 mitch    staff       686 Jun 13  2006 QDDS_ORB.tcsh*
drwxrwxr-x   2 mitch    staff      1024 Jun 14  2006 QDDS_orig/
-rw-rw-r--   1 mitch    staff       484 Jun 13  2006 README
-rwxrwxr-x   1 mitch    staff       669 Jun 14  2006 remove_old_diffpolldir.tcsh
```

3 4. Directory structure

There is a /home/qdds/ directory, which is where the data are managed. The listing of this directory is as follows with the most important files/directories highlighted:

```
-rw-rw-r--   1 nobody   nobody       91 May  7  2002 crontab
drwxrwsr-x   5 scott    analyst     512 Jul 18  2003 dbcron/
-rw-rw-r--   1 nobody   nobody      635 May  6  2002 newlog.sh
drwxrwsr-x   2 nobody   analyst     512 Sep  6  2005 polldir/
drwxrwsr-x   2 nobody   analyst     512 Jul 10 16:30 polldiralarm/
drwxrwsr-x   2 nobody   analyst    6144 Jul 10 16:30 polldirdb/
drwxrwsr-x   2 nobody   analyst    6144 Jul  9 16:35 polldirdelete/
drwxrwsr-x   2 nobody   analyst   19456 Jun 13  2006 polldirdiff/
drwxrwsr-x   2 nobody   analyst   18944 Jul 10 16:30 polldirfinish/
drwxrwsr-x   2 nobody   analyst     512 Jul 10 16:30 polldirorb/
drwxrwxr-x   7 nobody   nobody      512 May  6  2002 QDDS/
```

```
-rwxrw-r--   1 nobody   nobody       865 May  7  2002 QDDS.cronscript*
-rw-r--r--   1 nobody   nobody         0 Jul 10 16:49 QDDS.cronscript.log
drwxrwsr-x   2 scott    analyst      512 Apr 27  2006 qddslock/
drwxrwxr-x   3 nobody   nobody       512 Apr 16  2004 QDDS_new20040416/
drwxrwxr-x   7 nobody   nobody       512 Jan 19  2006 run/
drwxrwsr-x   2 nobody   analyst      512 May 14  2002 temppolldir/
drwxrwsr-x   2 nobody   analyst      512 Jul 17  2003 temppolldir_info/
drwxrwsr-x   2 nobody   analyst      512 Nov 18  2002 temppolldir_new/
drwxrwsr-x   2 nobody   analyst      512 Apr 27  2006 temppolldir_qdds/
drwxrwsr-x   2 nobody   analyst      512 Jul 22  2003 tmpdbdir/
```

We've already seen the run/ directory is the actual QDDS system. The QDDS/ and QDDS_new20040416/ are just the original downloads of the QDDS system from the ftp site.

The directory polldir/ is obsolete – only run/polldir/ is used now. The temppoll*/ directories are also obsolete as is tmpdbdir/ and polldirdiff/. The directory qddslock/ is empty.

The following directories seem to be updated daily:

polldiralarm/	empty most of the time?
polldirdb/	currently has 4 EVENT.????? files
polldirdelete/	lots of files like delete.?????
polldirfinish/	lots of files like EVENT.????? and event.?????
polldirorb/	just has two ORB.????? from 2006 – may be obsolete

The following file is also updated:

QDDS.cronscript.log

The shell script QDDS.cronscript just makes sure that the QDDS system (run/QDDS.jar) is running. Its run every half-hour and restarts QDDS if necessary.

The program newlog.sh just limits the sizes of logfiles and is designed to be run once a week, but doesn't appear to point to any thing of use, and is probably obsolete.

Then there is also /Seis/mitch/orb_scott_qdds/. This listing is:

```
-rw-------   1 scott    analyst  10780672 Jul  5  2004 core
-rw-r--r--   1 scott    analyst        38 Jul 10 16:25 createqddsdb.log
-rw-rw-r--   1 scott    analyst      9091 Jul  2 15:04 orb_qdds.mail
-rw-r--r--   1 scott    analyst      9091 Jul 10 15:07 orb_qdds.mail_old
-rw-rw-r--   1 scott    analyst      2217 Jul 10 15:07 orb_qdds.temp
drwxrwxr-x   2 scott    analyst       512 May 15  2002 polldir/
-rw-rw-r--   1 scott    analyst       250 Apr 10 11:07 QDDS_ORB.log
```

```
-rw-r--r--   1 scott    analyst  2963223 Jul 10 02:43 remove_old_diffpolldir.log
-rw-r--r--   1 scott    analyst   451717 Jul 16  2003 remove_old_orb.tcsh.log
drwxrwxr-x   2 scott    analyst     2048 May 16  2002 testdiffpolldir/
drwxrwxr-x   2 scott    analyst     2048 May 16  2002 testpolldir/
```

The only non-obsolete items here seem to be the logfiles highlighted:

createqddsdb.log	'resize: can't open terminal /dev/tty'
orb_qdds.mail	'starting up orb_qdds'
orb_qdds.mail_old	'starting up orb_qdds'
orb_qdds.temp	result of a 'ps -ax' command.
QDDS_ORB.log	list of what seem to be processes started in the background.
remove_old_diffpolldir.log	'/home/qdds/polldirfinish/event.????? removed'. Created by cronjob `remove_old_diffpolldir.tcsh`

Finally there are also many matches to /home/mitch/*/*qdds*/ and /home/mitch/*/*QDDS*/. I haven't tried to follow these, assuming them to be obsolete since they are in a users home directory.

4 5. Automatic submission

There are cronjobs running as user scott on segment.giseis.alaska.edu:

```
43 10 * * * /usr/local/bin/QDDS/remove_old_diffpolldir.tcsh >> /Seis/mitch/
orb_scott_qdds/remove_old_diffpolldir.log 2>&1
7 3,7,11,15,19,23 * * * /usr/local/bin/QDDS/QDDS_ORB.tcsh >> /Seis/mitch/
orb_scott_qdds/QDDS_ORB.log 2>&1
25 * * * * /usr/local/bin/QDDS/createqddsdb >
/Seis/mitch/orb_scott_qdds/createqddsdb.log 2>&1
```

5.1Automatic submission of new origins

The cronjob which calls QDDS_ORB.tcsh is the one which makes sure that new /db/origin packets which arrive on earlybird:6510 are submitted to QDDS.

QDDS_ORB.tcsh

Essentially this starts orb_qdds if it isn’t running, and its run every 4 hours. The command is:

```
nohup /usr/local/bin/QDDS/orb_qdds /home/qdds/run/polldir /home/qdds/polldirorb
earlybird >> /Seis/mitch/orb_scott_qdds/orb_qdds.mail
```

Logging information is saved to orb_qdds.mail and orb_qdds.temp, and the former is emailed to Mitch.

orb_qdds.c

Opens earlybird:6510 (orbopen).
Waits for a /db/origin packet (orbselect).
Goes into an infinite loop.

- Orbreap gets some packet information (orbreap).
- Unstuffs the packet. (unstuffPkt)
- Clears the register.
- Runs **dborigin_to_qdds** to generate '$aline' which describes the event in qdds format:
  rc = dborigin_to_qdds(unstuffed->db, aline, 0, 'A', &evid, 0, "origin")
- Sets $qddsfilename = run/polldir/orb$evid
- Sets $qddsfilenamediff = polldirorb/orb$evid
- Prints $aline to $qddsfilename and $qddsfilenamediff.

The crucial thing here is automatic events go into polldirorb/ as well as run/polldir/ (i.e. Direct to QDDS). So they first appear in the interface directories as orb.$evid files under polldirorb.

dborigin_to_qdds.c

This is pretty complicated so hopefully it doesn't need changing – its job is to generate an appropriate QDDS code encoded in the variable $aline.
Input arguments are $db, $aline, $add, $loc_method, $evid, $version and $joinviewname. So when called by orb_qdds.c, $add = 0, $version = 0 and $joinviewname = 'origin'.
Among the information it codes in $aline appears to be the closest station, the seismic gap, the number of stations used, the origin errors, a code saying whether its a new event, delete event request or update request, origin time and hypocentre and magnitude.

5.2 Automatic submission from the operational database

createqddsdb

This calls /usr/local/bin/db_quake_qdds 5 $diffpolldb $maindb. That is it calls:
db_quake_qdds 5 polldirdb/ /iwrun/op/db/seg
The upshot is that a new $aline (CUBE code) will be written for the preferred origin to polldirdb/event.$id, using a version of 5, if it can be found in /iwrun/op/db/seg/quakes.

It then looks if the /home/$user/alarm.origin table exists – but since this is an automatic update, there wont be an alarm database.

All files matching polldirorb/orb.* get moved to polldirfinish/.

All files matching polldirdb/event.* get moved to polldirfinish/ and run/polldir/ (unless polldirdelete/delete.* exists).

All files matching polldiralarm/event.* get moved to polldirfinish/alarm.* and run/polldir/alarm.* (unless polldirdelete/delete.* exists).

db_quake_qdds

Sets $joinviewname to 'qddsjoin'. Command line arguments are $version, $polldir and $db_name.

Opens $db_name and joins the event and origin tables and subsets for 'prefor == orid' and 'evid > 1000' if the event table has records, and stores this with pointer $dbevent and calls the view $joinviewname. Otherwise it justs sets $dbevent to the origin table.

Then it loops over all events and calls dborigin_to_qdds (see section 5.1) with arguments $dbevent, $aline, 0, 'A', &evid, $version (5 or 6 here) and $joinviewname ('qddsjoin'). The return code ($rc) is checked, and if its zero (i.e. OK) it sets $qddsfilename to $polldir/event.$id and writes $aline to it (which has come from dborigin_to_qdds).

Finally it closes the databases and returns 1.

5.3Removal of old QDDS events

This cronjob is run once a day and simply removes any files matching polldirfinish/event.*.

remove_old_diffpolldir.tcsh

Removes files older than 10 days. Uses the useful script /usr/tools/scripts/file_age.

5 6. Manually deleting events using dbevents

dbevents

The beeper duty uses the alias duty_dbevents to fire up an AEIC version of dbevents (see manual on internal webpage) to examine recent automatic events, and if necessary removes them from QDDS.

When a user right-clicks on an origin on the map, it will bring up a menu which includes the option to delete from QDDS. Clicking on this menu list item calls the *delete_event_qdds* routine.

delete_event_qdds **(subroutine in dbevents)**

1. sets up the globals: $delete_evid, $evtime, $dbname, $qddson and $Pf
2. sets $qdds_mail_list (from $Pf), $qdat (date from $evtime), $yy, $mo, $dy2 (year, month, day from $qdat), and $qdate (from $yy, $mo, $dy2)
3. Asks ‘are you sure?’
4. it checks if $qddson == 1, if not it display error ‘Permission to delete from QDDS not allowed, See Control pulldown menu to allow permission’
5. Execs **DELETE_QDDS_EVENTID $qdate $delete_evid**
6. it checks if /home/qdds/polldirdelete/delete.$delete_evid exists

7. if it does, it displays the message ‘Deleted $evid from QDDS submission’, and then calls *send-qdds_email* to $qdds_mail_list $delete_evid
8. if it doesn’t it displays an error message ‘AEVENT.evid does not exist, perhaps createqddsdb is still running’

send_qdds_email **(subroutine in dbevents)**

The (tk/tcl) routine *send_qdds_emai*l just composes an email from origin information using rtmail.

DELETE_QDDS_EVENTID

1. sets $diffpollorb, $diffpolldb, $diffpollalarm, $diffpolldelete and $diffpollfinish
2. sets $polldir = 'run/polldir'
3. sets $curtime = $qdate and $evid = $delete_evid (from command line)
4. checks if polldirdelete/delete.$evid exists
5. if it does, it checks for the following files in descending priority:

- polldirfinish/alarm.$evid (manually generated from an alarm release, and moved by createqddsdb)
- polldirfinish/event.$evid (automatically generated from the database, and moved by createqddsdb)
- polldirfinish/orb.$evid (automatically generated from the orb, and moved by createqddsdb)
- polldiralarm/event.$evid (manually generated from an alarm release)
- polldirdb/event.$evid (automatically generated from the database)
- polldirorb/orb.$evid (automatically generated from the orb)

6. $deletefile is set to the highest priority of these.
7. if $deletefile set, it then calls **delete_qdds_eventid $curtime $deletefile $diffpolldelete** (remember $curtime = $qdate from dbevents).
8. checks for a file matching polldirdelete/*.$evid. if its there copies it to run/polldir/*.$evid and renames it to polldirdelete/delete.$evid

This last step seems to be crucial: An event marked for deletion seems to be placed at polldirdelete/event.$evid by delete_qdds_eventid. It then gets copied to run/polldir/ and then renamed to polldirdelete/delete.$evid.

What I don't know is how do events get into polldirdb/ and polldiralarm/. I do know that they get from those and polldirorb/ into polldirfinish/ via createqddsdb which is invoked through running wormwatch to release an event.

delete_qdds_eventid.c

1. sets $iyear, $imonth and $iday from $curtime (=$qdate)
2. opens $filename for reading (this is the file selected for deletion from the priority list)
3. scans $filename for $aline = type, evid, source, version, year, month, day, hour, minute, sec, lat, lon, depth, mag, nst, ndef etc...
4. checks if this year, month and day match $iyear, $imonth and $iday, and if source=’AK’ and type = ‘E ‘.
5. Seems to replace type with ‘DE’ in $aline.
6. Sets $qddsfilename = polldirdelete/del$version.$evid (version = 0**)**
7. Writes new $aline to $qddsfilename

8. $filename2 = toupper($filename)
9. mv $filename $filename2

So in short this program seems to add the line $aline with E changed to DE to the file polldirdelete/del$version.$evid, and it seems to rename the $deletefile to toupper($deletefile).

6 7. Manual updates from dbe

dbe

Although Antelope 4.9 has been installed, the current default at AEIC is version 4.8 at /opt/antelope/4.8/bin/dbe. This is effectively an alias to /opt/antelope/4.8/data/tcl/library/dbe/startup.

Curiously dbe uses a parameter file called .dbe.pf. The default is located at /opt/antelope/4.8/data/pf/.dbe.pf. Among other things it defines the menu items that will show up when a user attempts to edit an origin table. At AEIC this default (and others) are overriden by using a system-wide setup that references /usr/local/aeic/4.8/data/pf/.dbe.pf also. This leads dbe to create widgets on the edit menu when a user looks at an origin table called 'Update' and 'Respond' which call 'aeic_update_location' and 'aeic_respond' respectively.

The “Update” option:

Clicking the update option calls **/opt/local/aeic/4.8/bin/aeic_update_location**.
This uses /usr/local/aeic/4.8/data/pf/aeic_release.pf from which it gets the %Helpers hash to allow it to call **/usr/local/aeic/4.8/bin/aeic_partial_release**. When aeic_partial_release runs it creates a widget 'update_qdds' which the duty person uses to call **/usr/local/bin/QDDS/createqddsdb**, again using the %Helpers hash from aeic_release.pf.

The “Respond” option:

Clicking the respond option calls **/opt/local/aeic/4.8/bin/aeic_respond**.
This uses /usr/local/aeic/4.8/data/pf/aeic_release.pf from which it gets the %Helpers hash to allow it to call **/usr/local/aeic/4.8/bin/aeic_release_distributor**. When aeic_release_distributor runs it creates a widget 'update_qdds' which the duty person uses to call **/usr/local/bin/QDDS/createqddsdb**, again using the %Helpers hash from aeic_release.pf.

createqddsdb

As we've already seen, this calls:
db_quake_qdds 5 polldirdb/ /iwrun/op/db/seg
The upshot is that a new $aline will be written for the preferred origin to polldirdb/event.$id, using a version of 5.

It then looks if the /home/$user/alarm.origin table exists – it will since this is a manual release – and calls:
db_quake_qdds 6 polldiralarm/ $HOME/alarm
The upshot is that a new $aline will be written for the preferred origin to polldiralarm/event.$id, using a version of 6.

All files matching polldirorb/orb.* get moved to polldirfinish/.

All files matching polldirdb/event.* get moved to polldirfinish/ and run/polldir/ (unless polldirdelete/delete.* exists).

All files matching polldiralarm/event.* get moved to polldirfinish/alarm.* and run/polldir/alarm.* (unless polldirdelete/delete.* exists).

***Note:* we can see from these steps that its by checking if polldirdelete/delete.$id that the Antelope interface to QDDS knows not to broadcast an event or alarm deleted earlier using dbevents_aeic.**