

SAM file delivery comparison between SAM V6 and SAM V7

Tests to measure the SAM file delivery performance were performed on the CDF phase I SAM farm. Python scripts were written to mimic the SAM behavior of a “typical” user’s job on the CDF CAF’s. The scripts are located in appendix A. CDF’s use of SAM differs from that of the D0 experiment. CDF uses the sam station to deliver the file URL and then uses the CDF offline analysis framework to open the file and read it in using the dcache system.

Some of the specifics of the tests are listed in table 1. The output gzipped for the V6 tests are in the directory: Fcdflnx3.fnal.gov:/cdf/scratch/cdfopr/mimic-cafm. The V7 test results are in the directory: fcdflnx3.fnal.gov:/cdf/scratch/cdfopr/mimic-cafk.

Table 1 List of SAM file delivery tests performed

	Submit time	End time	# of jobs & sections/job	# files/job (total files)	station name (version)	DB server (version)
V6	Wed Sep 7 13:05:58 2005	Wed Sep 7 17:10:10 2005	10 jobs - 8 sections	250 files/sect 2000 files/job	Cdf-dcache-teststand (v6_0_1_12)	User_int_old (v6_7_1_0)
V7	Tue Sep 6 10:10:55 2005	Tue Sep 6 12:09:56 2005	8 jobs – 50 sections	40 files/ sect 2000 files/job	Cdf-int (v6_0_2_5)	User_int (v7_3_0)

Table 2 lists the significant results of the testing; namely the number of files delivered per minute, number of files delivered per section, time per file, number of file not delivered.

Table 2 Summary of SAM file delivery test results

	File delivery rate (files/min)	File delivery rate (files/sec)	Time to deliver a file (sec/file)	# sections non-delivered files	Total non-delivered files
V6	86.32 files/min	1.44 files/sec	0.70 sec/file	12 sect – all jobs	45 (0.225%)
V7	178.6 files/min	2.98 files/sec	0.34 sec/file	0	0

It is clear to see from the comparison of the SAM V6 vs SAM V7 file delivery results. V7 SAM is significantly superior to V6 SAM in terms of file delivery time and file delivery accuracy. The next sections present plots to show these results.

Section 1 - SAM V6 Results:

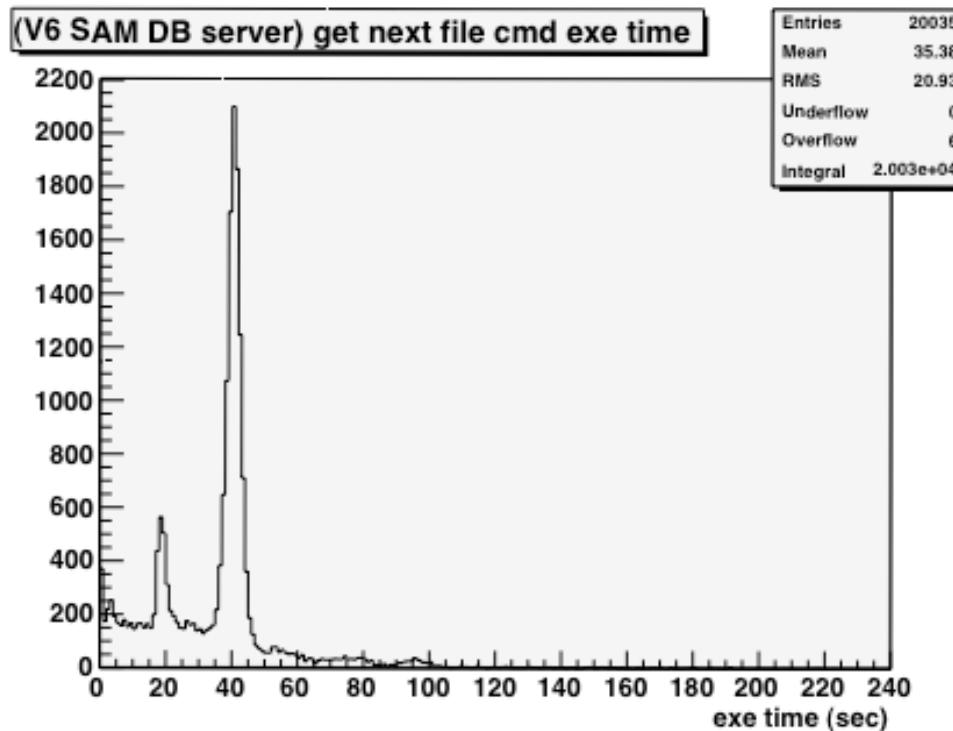


Figure 1 - Distribution of command execution time for get next file command for the SAM V6 configuration. The first peak near 20 section is before the station/db server become saturated. The second peak at ~40 sec occurs after the station/db server is saturated.

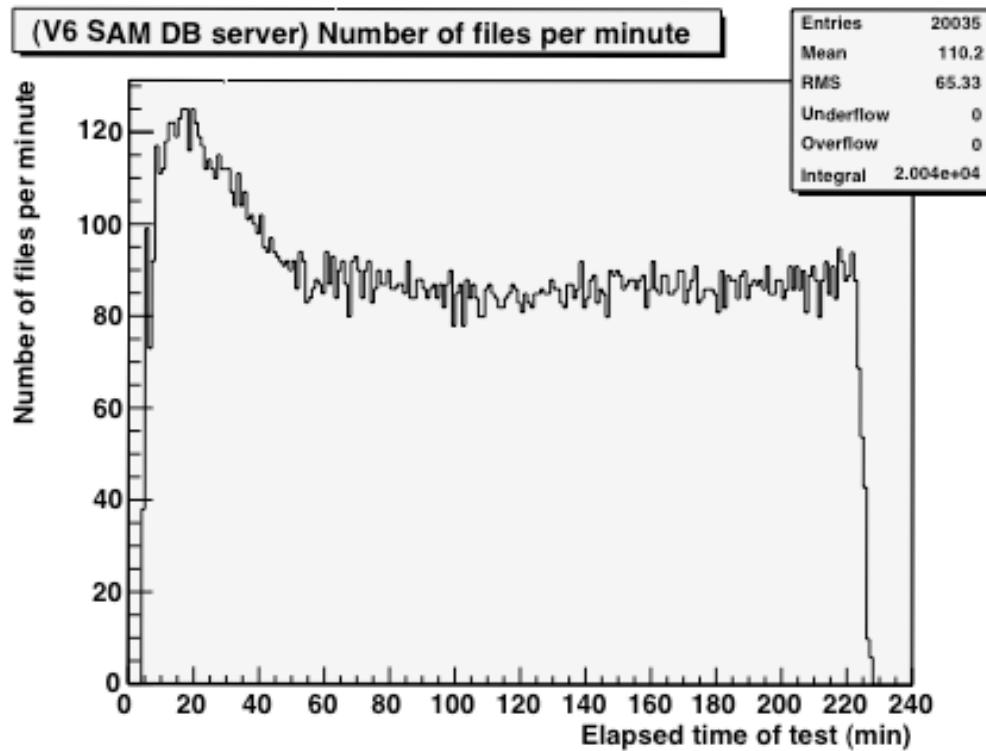


Figure 2 - Distribution of the number file URL deliveries per min for the SAM V6 configuration. The plateau indicates saturation of the station/db server.

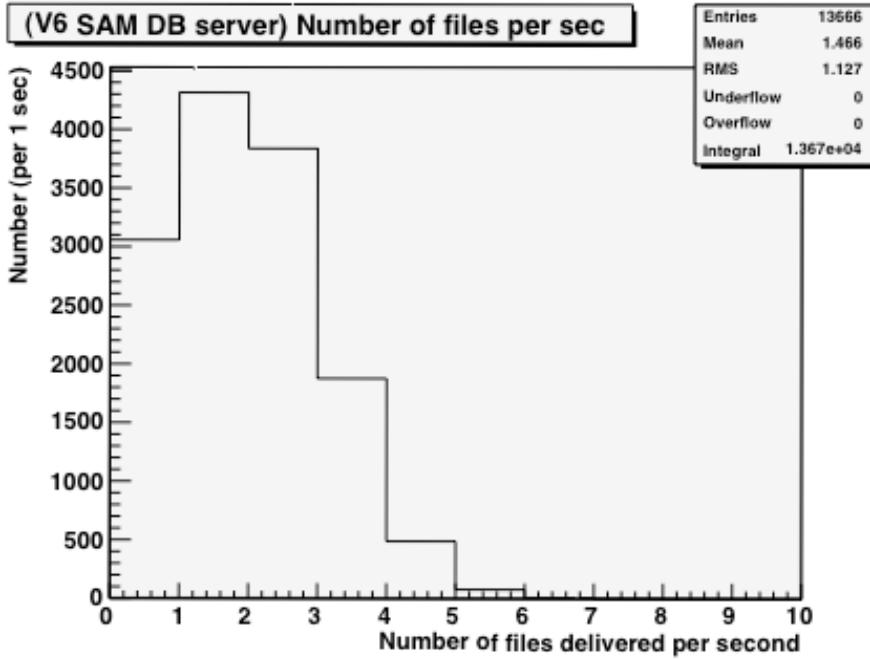


Figure 3 - Distribution of the number of file URL's delivered per second for the SAM V6 configuration. The mean is ~ 1.5 files per second.

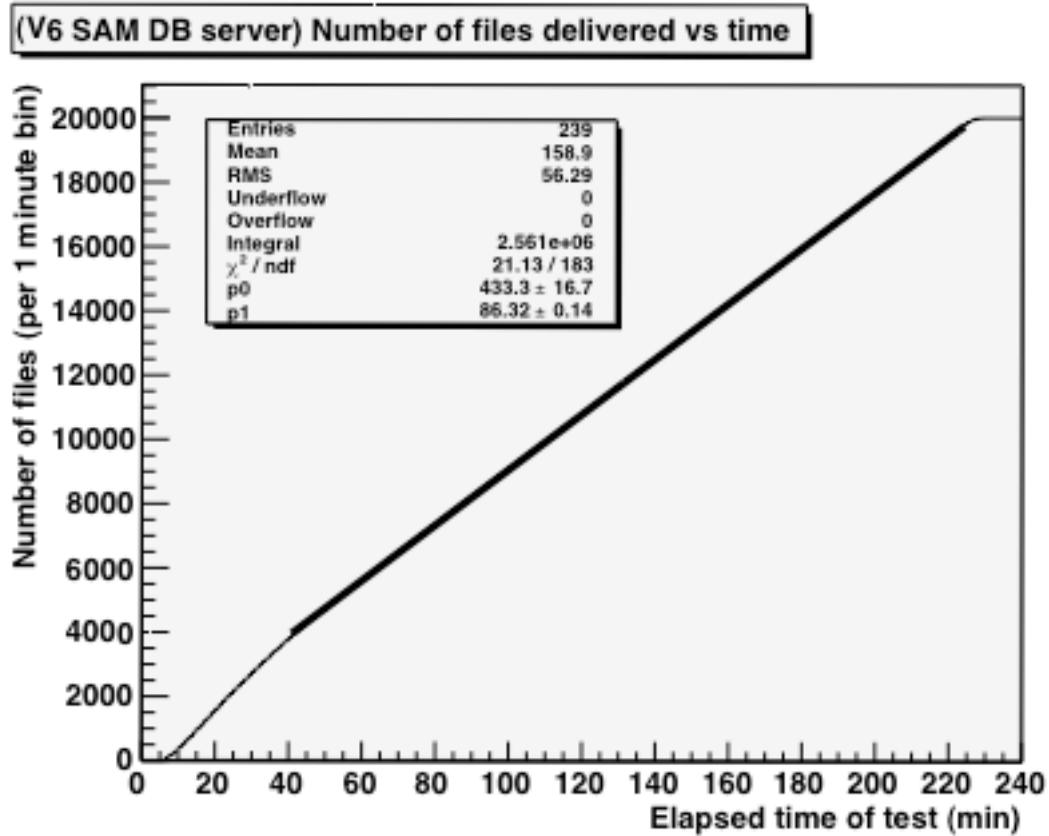


Figure 4 – Plot of the file delivery rate for the duration of the test for the V6 configuration. The station/DB server saturation is manifested by the knee at ~ 40 minutes into the test. The fit gives the number of files delivered per minute and is summarized in table 2.

Section 1 - SAM V7 Results:

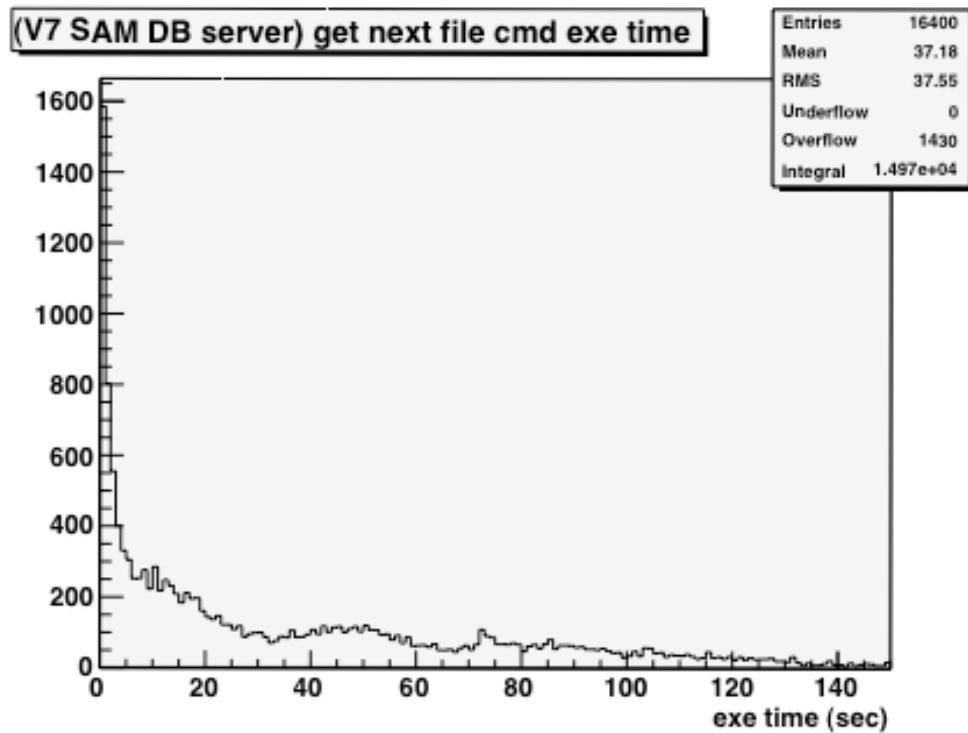


Figure 5 - Distribution of command execution time for get next file command for the SAM V7 configuration. The average command length of 37.17 seconds implies that SAM can not be used as part of interactive analysis of root ntuple files at CDF.

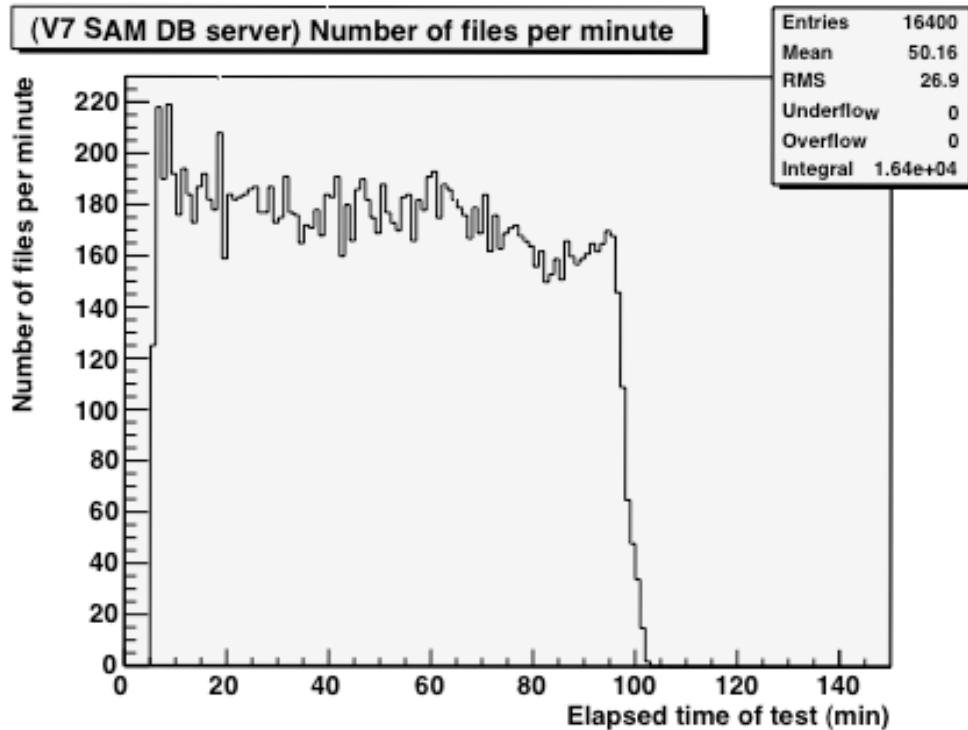


Figure 6 - Distribution of the number file URL deliveries per min for the SAM V7 configuration. The plateau indicates a saturation of the SAM station /db server.

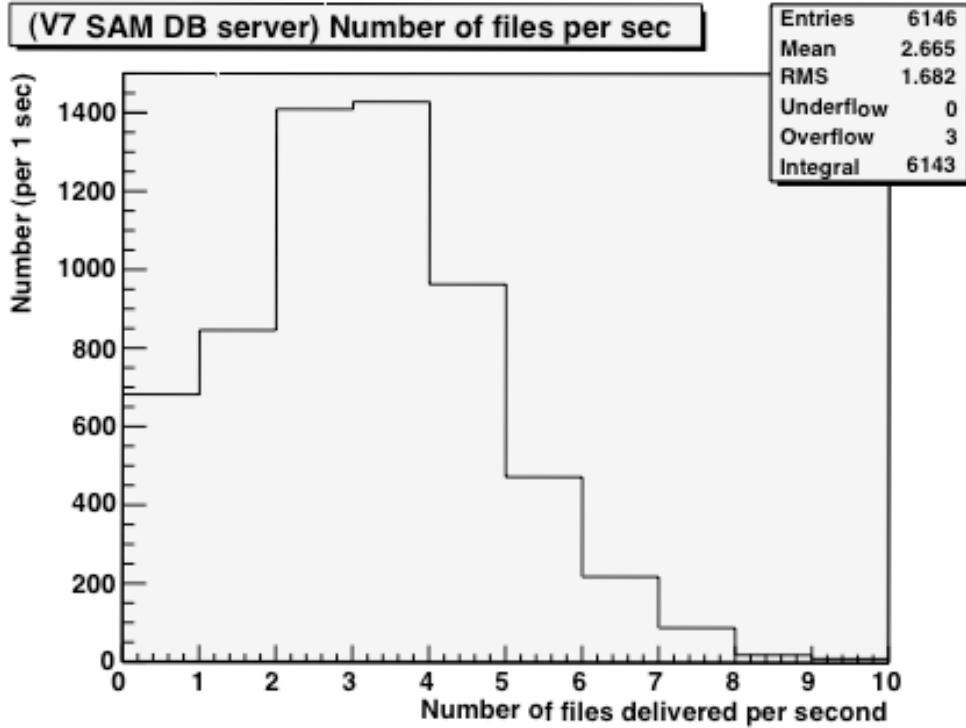


Figure 7 - Distribution of the number of file URL's delivered per second for the SAM V7 configuration. The mean is ~ 2.7 files per second.

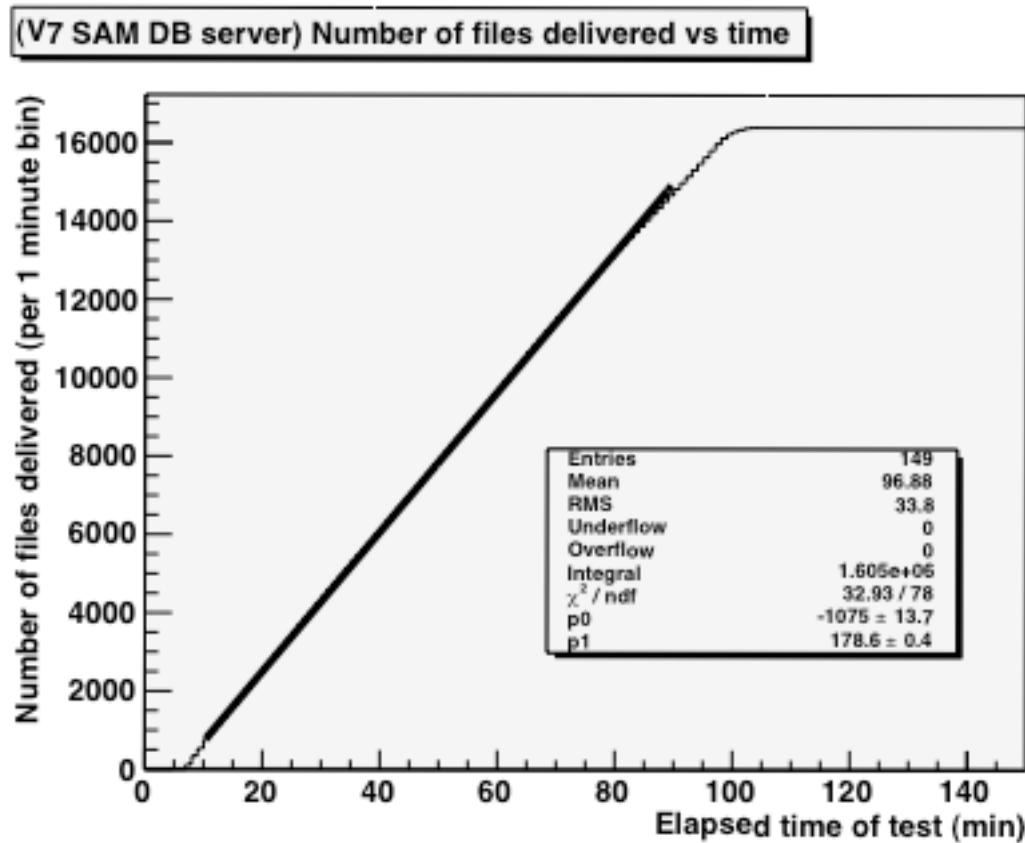


Figure 8 – Plot of the file delivery rate for the duration of the test for the V7 configuration. The fit gives the number of files delivered per minute and is summarized in table 2.

Conclusions:

These tests show a clear and marked improvement between the SAM V6 and SAM V7. It remains to be seen if a file delivery rate of almost 3 files/ second will be sufficient for CDF's usage on the FNAL CAF's. It is clear that much more work is needed before one can consider using SAM to deliver files during root ntuple processing.

Appendix A: Python source code used in test.

V6 python source code:

```
#!/usr/bin/env python
#
#      dpb-samgetdbserverConnectionInfo-v7
#
#      Get the dbserverConnectionInfo for the db server setup in the setup
sam -q ***** command
#
import string
import sys
import os
import getopt
import commands
import time

from sam import sam
from SamFile.SamDataFile import SamDataFile
from SamException import SamExceptions
from SamStruct.SamBoolean import SamBoolean

import stat
from time import gmtime, strftime, localtime

#
# first check that sam environment has been setup
#
job_start=time.time()
job_starttime=time.clock()
job_start_string = '%s' % strftime("%Y-%m-%d %H:%M:%S",
localtime(job_start))

print 'Job looping through events started: %s  ' % (job_start_string)

try:
    sam2 = os.environ['SETUP_SAM']
except:
    print "Error: sam not set. 'setup sam' before running me"
```

```

    sys.exit(1)

dbservername = os.environ[ 'SAM_DB_SERVER_NAME' ]

job_elapsed_time = 0


sam_station=os.environ['SAM_STATION']
sam_project=os.environ['SAM_PROJECT']
sam_dataset=os.environ['SAM_DATASET']

#
#  parse the options
#
file_limit=10

try:
    optlist, args = getopt.getopt(sys.argv[1:], 'f', ['file_limit='])
except getopt.GetoptError, e:
    sys.exit(1)

for key, value in optlist:
    if key == '--file_limit':
        file_limit=long(value)

print 'Set file limit to %d ' %(file_limit)

print "Started project: " + sam_project
print "on sam station : " + sam_station
print "using dataset : " + sam_dataset

#
# Establish the consumer
=====
#



print 'Started project: %s    on sam station : %s    using db server : %s
and dataset : %s    at %s '
%(sam_project,sam_station,dbservername,sam_dataset,job_start_string)

cid = 0

try:
    starttime=time.clock()
    task_start=time.clock()
    cid =
    sam.establishConsumer(project=sam_project,station=sam_station,appname='dem
o',appversion='1')
    stoptime_consumer=time.clock()
    elapsed_time_consumer = stoptime_consumer-starttime

```

```

starttime_string = '%s' %strftime("%Y-%m-%d
%H:%M:%S",localtime(task_start))
print 'Established Consumer ID#: %d at %s - it took %.3f secs to
run command' %(cid,starttime_string,elapsed_time_consumer)
except Exception, e:
    samStopStatus="SAM establish consumer failed: \n Exception:" +
str(e.__class__)+"\n"+str(e)
except:
    samStopStatus="SAM establish consumer failed: \n Unexpected,
unidentified exception"

#
# Establish the consumer process
=====
#
print cid
if cid > 0:
    try:
        task_start=time.time()
        starttime=time.time()
        cpid =
sam.establishProcess(project=sam_project,station=sam_station,cid=cid,limit
=file_limit)
        stoptime=time.time()
        elapsed_time = stoptime-starttime
        starttime_string = '%s' %strftime("%Y-%m-%d
%H:%M:%S",localtime(task_start))
        print 'Established Consumer Process CPIID = %d at %s - it took
%.3f secs to run command' %(cpid,starttime_string,elapsed_time)

    except Exception, e:
        samStopStatus="SAM establish consumer process failed: \n
Exception:" + str(e.__class__)+"\n"+str(e)
    except:
        samStopStatus="SAM establish consumer process failed: \n
Unexpected, unidentified exception"

#
# Set the file limit for each consumer process
=====
#
try:
    task_start=time.time()
    starttime=time.time()
    result =
sam.increaseDeliveryLimit(processId=cpid,project=sam_project,station=sam_s
tation,limit=file_limit)
    stoptime=time.time()
    elapsed_time = stoptime-starttime
    starttime_string = '%s' %strftime("%Y-%m-%d
%H:%M:%S",localtime(task_start))
    print 'Set file limit to %d at %s - it took %.3f secs to run command'
%(file_limit,starttime_string,elapsed_time)
except Exception, e:

```

```

    samStopStatus="SAM establish consumer failed: \n Exception:" +
str(e.__class__)+"\n"+str(e)
except:
    samStopStatus="SAM establish consumer failed: \n Unexpected,
unidentified exception"

#
=====
=====
# and here is the loop over the files
#
=====
=====

file_number=0
while 1:
    task_start=time.time()
    starttime = time.time()
    the_file =
sam.getNextFile(project=sam_project,station=sam_station,cpid=cpid)
    stoptime=time.time()
    elapsed_time = stoptime-starttime
    starttime_string = '%s' %strftime("%Y%m%d
%H%M%S",localtime(task_start))
    print 'getNext_file %s %.3f' %(starttime_string,elapsed_time)
    if the_file == '' or the_file == 'END' or the_file == 'ERROR' or
the_file == 'END OF STREAM' :
        break
    file_number = file_number + 1
    print "Got file ", file_number , " ", the_file

# now sleep for 15 seconds

    time.sleep(15)

    task_start=time.time()
    starttime = time.time()
    sam.releaseFile(project=sam_project,station=sam_station,cpid=cpid,fil
ename=the_file,status='ok')
    stoptime=time.time()
    elapsed_time = stoptime-starttime
    starttime_string = '%s' %strftime("%Y%m%d
%H%M%S",localtime(task_start))
    print 'release_file %s %.3f' %(starttime_string,elapsed_time)

#
# get the total elapsed time etc.
#
job_stop=time.time()
job_stoptime=time.time()

job_stoptime_string = '%s' %strftime("%Y-%m-%d
%H:%M:%S",localtime(job_stop))

```

```

job_elapsed_time = job_stop-job_start

print 'mimic-caf-usage finished at %s - it took %.3f secs to run python
script ' %(job_stoptime_string,job_elapsed_time)

```

V7 python source code:

```

#!/usr/bin/env sampy
#
#      dpb-samgetdbserverConnectionInfo-v7
#
#      Get the dbserverConnectionInfo for the db server setup in the setup
sam -q ***** command
#

import string
import sys
import os
import getopt
import commands
import time
from Sam import sam
from SamFile.SamDataFile import SamDataFile
from SamException import SamExceptions
from SamStruct.SamBoolean import SamBoolean
from SamStruct.DbServantConnectionInfoList_v2 import
DbServantConnectionInfoList_v2

import stat
from time import gmtime, strftime, localtime

#
# first check that sam environment has been setup
#
job_start=time.time()
job_starttime=time.clock()
job_start_string = '%s' % strftime("%Y-%m-%d %H:%M:%S",
localtime(job_start))

print 'Job looping through events started: %s  % (job_start_string)

try:
    sam2 = os.environ['SETUP_SAM']
except:
    print "Error: sam not set. 'setup sam' before running me"
    sys.exit(1)

dbservername = os.environ['SAM_DB_SERVER_NAME']

job_elapsed_time = 0

```

```

sam_station=os.environ['SAM_STATION']
sam_project=os.environ['SAM_PROJECT']
sam_dataset=os.environ['SAM_DATASET']

#
#  parse the options
#
file_limit=10

try:
    optlist, args = getopt.getopt(sys.argv[1:], 'f', ['file_limit='])
except getopt.GetoptError, e:
    sys.exit(1)

for key, value in optlist:
    if key == '--file_limit':
        file_limit=long(value)

#file_limit=long(os.environ['SAM_FILE_LIMIT'])

#
# Establish the consumer
=====
#
print 'Started project: %s  on sam station : %s  using db server : %s
and dataset : %s  at %s '
%(sam_project,sam_station,dbservername,sam_dataset,job_start_string)

try:
    starttime=time.clock()
    task_start=time.clock()
    cid =
    sam.establishConsumer(project=sam_project,station=sam_station,appname='dem
o',appversion='1')
    stoptime_consumer=time.clock()
    elapsed_time_consumer = stoptime_consumer-starttime
    starttime_string = '%s'  %strftime("%Y%m%d
%H%M%S",localtime(task_start))
    print 'Established_Consumer %d %s %.3f'
    %(cid,starttime_string,elapsed_time_consumer)
except Exception, e:
    samStopStatus="SAM establish consumer failed: \n Exception:" +
    str(e.__class__)+"\n"+str(e)
except:
    samStopStatus="SAM establish consumer failed: \n Unexpected,
unidentified exception"

#
# Establish the consumer process
=====
```

```

#
if cid > 0:
    try:
        task_start=time.time()
        starttime=time.time()
        cpid =
sam.establishProcess(project=sam_project,station=sam_station,cid=cid,limit
=file_limit)
        stoptime=time.time()
        elapsed_time = stoptime-starttime
        starttime_string = '%s' %strftime("%Y%m%d
%H%M%S",localtime(task_start))
        print 'Established_Consumer_Process %d %s %.3f '
%(cpid,starttime_string,elapsed_time)

    except Exception, e:
        samStopStatus="SAM establish consumer process failed: \n
Exception:" + str(e.__class__)+"\n"+str(e)
    except:
        samStopStatus="SAM establish consumer process failed: \n
Unexpected, unidentified exception"

#
# Set the file limit for each consumer process
=====
#
# section removed
#
=====
=====

# and here is the loop over the files
#
=====
=====

file_number=0
while 1:
    task_start=time.time()
    starttime = time.time()
    the_file =
sam.getNextFile(project=sam_project,station=sam_station,cpid=cpid)
    stoptime=time.time()
    elapsed_time = stoptime-starttime
    starttime_string = '%s' %strftime("%Y%m%d
%H%M%S",localtime(task_start))
    print 'getNext_file %s %.3f' %(starttime_string,elapsed_time)
    if the_file == '' or the_file == 'END' or the_file == 'ERROR' or
the_file == 'END OF STREAM' :
        break
    file_number = file_number + 1
    print "Got file ", file_number , " ", the_file

# now sleep for 15 seconds

```

```

time.sleep(15)

task_start=time.time()
starttime = time.time()
sam.releaseFile(project=sam_project,station=sam_station,cpid=cpid,fil
ename=the_file,status='ok')
stoptime=time.time()
elapsed_time = stoptime-starttime
starttime_string = '%s' %strftime("%Y%m%d
%H%M%S",localtime(task_start))
print 'release_file %s %.3f' %(starttime_string,elapsed_time)

#
# check that file_number = file_limit
#
if file_number < file_limit:
    print 'file_number < file_limit %d %d' %(file_number,file_limit)

if file_number > file_limit:
    print 'file_number > file_limit %d %d' %(file_number,file_limit)

#
# get the total elapsed time etc.
#
job_stop=time.time()
job_stoptime=time.time()

job_stoptime_string = '%s' %strftime("%Y%m%d %H%M%S",localtime(job_stop))
job_elapsed_time = job_stop-job_start

print 'mimic-caf-usage finished at %s - it took %.3f secs to run python
script' %(job_stoptime_string,job_elapsed_time)

```