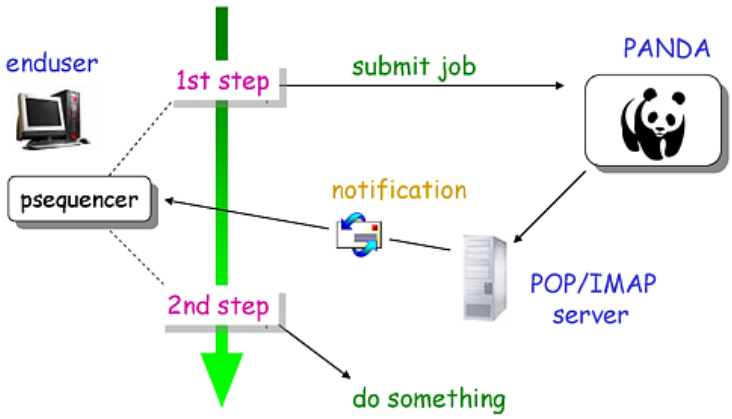
Table of Contents

How to perform sequential jobs/operations in PanDA based analysis

Introduction

psequencer is an email-driven application which executes a group of processes seqentially. Panda sends email-notifications as soon as user jobs are completed. Sometimes people may want to start something (e.g., downloading output files or submitting another job) automatically when they receive notifications. psequencer uses the notifications to trigger subsequent processes.



Getting started

Setup

psequencer is included in the panda-client package. See Installation and Setup for panda-client.

psequencer requires one configration file, ~/.pathena/panda.cfg, which contains how to connect to your email server. e.g.,

```
$ cat ~/.pathena/panda.cfg
[sequencer]
mail_protocol = pop3
mail_host = email.britannia.vi
mail_port = 995
mail_user = lelouch
mail_pass = Qa%hcX45b0_$
mail_ssl = True
mail_interval = 5
mail_keepalive = False
```

See Options in panda.cfg for detail.

 Make sure that you don't **physically** delete the email notifications from the mail server immediately. Especially if you are using POP, see Tips for POP users. In IMAP, 'delete' moves the mail to the trash folder or somewhere, and it will be automatically purged by the server after several days. As long as the mail exists in some folder on the server, psequencer should find it.

How to run

One sequence is composed of multiple steps, and each step executes one or more commands. For example, a common usecase could be that

1. pathena submits a job,
2. and then dq2-get retrieves output

where 1 and 2 are steps, and 1+2 is a sequence. The sequence needs to be written in a file,

```
$ cat myseq.txt

### Step1
cd ~/cmthome
source setup.sh -tag=15.6.10,32,setup
cd ~/myWork/run
pathena -v jobOptions.pythia14.py --outDS user.tmaeno.testEvt

### Step2
source /afs/cern.ch/atlas/offline/external/GRID/ddm/DQ2Clients/setup.sh
dq2-get user.tmaeno.testEvt

### Sequence
Step1.execute()
result = Step1.result()
if result.status == 0:
    Step2.execute()
```

How to run is simply

```
$ psequencer myseq.txt
```

or

```
$ psequencer -v myseq.txt
```

if you want to see what's going on.

One sequence file must contain some step sections followed by one sequence section. One step section starts with `### XYZ`, where `XYZ` is an arbitrary name except `Sequence` which is reserved for sequences. Each step can have multiple commands executed at the same time, e.g.,

```
### aStep
date
mv aho.dat aho.dat.back
echo aho.dat | root.exe
```

where three commands are executed in the step.

The sequence section starts with `### Sequence` and defines how steps are processed. The definition is written in the regular python syntax. In the sequence section, each step is executed by the `execute()` method and the result is available by using the `result()` method. The result has the status code (`status`) and the stdout (`output`). In addition, if the step runs a pathena job the following attributes are available;

```
JobID
Site
Jobs
Succeeded
Partial
Failed
In
Out
```

For example,

```
Summary of JobID : 123

Created : 2008-08-23 03:31:47 (UTC)
Ended   : 2008-08-23 14:41:24 (UTC)

Site    : ANALY_LONG_BNL_ATLAS

Total Number of Jobs : 800
           Succeeded : 681
           Partial   : 100
           Failed    : 19

In  : user08.KeteviAAssamagan.lxplus213_54.lib._000070
In  : user08.KeteviAAssamagan.005008.CavernInputSF05.simul.pool.v18
In  : ddo.000001.Atlas.Ideal.DBRelease.v050601
Out : user08.KeteviAAssamagan.005008.CavernInputSF05.RDO.pool.v3
```

is mapped to

```
result.JobID     = 123
result.Site      = 'ANALY_LONG_BNL_ATLAS'
result.Jobs      = 800
result.Succeeded = 681
result.Partial   = 100
result.Failed    = 19
result.In        = ['user08.KeteviAAssamagan.lxplus213_54.lib._000070','user08.KeteviAAssamagan.0
result.Out       = ['user08.KeteviAAssamagan.005008.CavernInputSF05.RDO.pool.v3']
```

They can be used in the sequence section for condition checking, e.g.,

```
### Sequence

nTry = 3
while nTry > 0:
    AOD_Step.execute()
    resAOD = AOD_Step.result()
    if resAOD.status == 0 and resAOD.Failed == 0:
        break
    nTry -= 1

DPD_Step.execute()
resDPD = DPD_Step.result()
if resDPD.status == 0 and resDPD.Succeeded > 10:
    DQ2_Step.execute()
    ROOT_Step.execute()
```

The above example tries an AOD job three times at most, and submits a DPD job. Then if the number of succeeded sub-jobs for DPD making is greater than 10, the DPD will be downloaded and a ROOT job will run locally on the DPD.

Tips

How to use running/frozen Panda jobs in a Sequence

Sometimes you may want to use Panda jobs which were already submitted outside of psequencer. In this case, you don't need to define a job submission step. You can get a running/frozen job, all running jobs, all frozen jobs by using `getPandaJob(JobID)`, `getRunningPandaJobs()`, and `getFrozenPandaJobs()`, respectively. All functions and attributes of Step, such as `result()` and `Succeeded`, are available except `execute()` which is not required of course.

```
### Sequence
job = getPandaJob(123)
res = job.result()
print res.status

rjobs = getRunningPandaJobs()
for tmpjob in rjobs:
    tmpres = tmpjob.result()
    print tmpres.status

fjobs = getFrozenPandaJobs()
for tmpjob in fjobs:
    tmpres = tmpjob.result()
    print tmpres.status
```

Note that `getFrozenPandaJobs()` returns a list of jobs frozen in the last 3 days by default. If you are interested in jobs older than 3 days, set a larger value to `--initScanDepth`. e.g.,

```
$ psequencer --initScanDepth 14 ...
```

How to give parameters to a Step

You can set the `env` argument in the `execute()` method to give parameters to the step. E.g.,

```
### StepXX
source /afs/cern.ch/atlas/offline/external/GRID/ddm/DQ2Clients/setup.sh
dq2-get -f $OUTFILE $OUTDATASET

### Sequence
var = {}
var['OUTFILE']    = 'AOD.073447._000111.pool.root.1'
var['OUTDATASET'] = 'data09_cvalid.00000001.SitesValidation.recon.AOD.o4_r733_tid073447'
StepXX.execute(env=var)
```

Internally the step is executed as

```
#!/bin/bash
export OUTFILE='AOD.073447._000111.pool.root.1'
export OUTDATASET'='data09_cvalid.00000001.SitesValidation.recon.AOD.o4_r733_tid073447'
source /afs/cern.ch/atlas/offline/external/GRID/ddm/DQ2Clients/setup.sh
dq2-get -f $OUTFILE $OUTDATASET
```

How to get result of Panda step/job without blocking

By default when the `result()` method is invoked it will block until the result is available. When it is invoked with the blocking argument set to False, it will immediately return None if the result is unavailable yet.

```
while True:
```

```
res = step.result(blocking=False)
if res != None:
    break
time.sleep(300)
```

Details

Options in panda.cfg

~/.pathena/panda.cfg is a configration file for psequencer. It contains one section led by `[sequencer]` and followed by `name=value` entries. Explanation of each entry is as follows;

mail_protocol
: protocol to connect to your mail server (pop3 or imap)

mail_host
: the host name of the mail server

mail_port
: the port number of the mail server

mail_user
: your user ID to logon to the mail server

mail_pass
: your password to logon to the mail server

mail_ssl
: True if using SSL (e.g. POP3/IMAP4 over SSL). Otherwise, False

mail_interval
: How frequently access to the mail server (in minute)

mail_keepalive
: False unless the mail server supports streaming and/or push-mail

Here is an example using Gmail IMAP.

```
[sequencer]
mail_protocol = imap
mail_host = imap.gmail.com
mail_port = 993
mail_ssl = True
mail_user = Lelouch.Lamperouge@gmail.com
mail_pass = $1mb@bWe
mail_interval = 5
mail_keepalive = False
```

Tips for POP users

Although IMAP is much better than POP in general, some people may still be using POP. Normally, mail clients delete messages from the remote POP server after they have retrieved the messages. psequencer tries to find notifications in the POP server, so the notifications need to stay at the server for a while. You can set recent email clients to leave a copy of mails at the mail server for several days. Make sure that you don't get the POP server clogged up when using those options.

Outlook Express

1. Select Tools | Accounts... from the menu in Windows Mail or Outlook Express.
2. Highlight the desired email account.
3. Click Properties.
4. Go to the Advanced tab.
5. Make sure Leave a copy of messages on server is selected.
6. Now also check Remove from server after __ day(s).
7. Specify how long you want messages to be kept at the server, basically 1 day is enough

fetchmail

1. Add `keep` and `uidl` to ~/.fetchmailrc

However, fetchmail is not so smart, AFAIK. The above setting doesn't delete messages at all, so the POP server will be filled up eventually. Another solution could be to forward email notifications to an IMAP server, e.g.,

1. Get a Gmail account at http://mail.google.com/
2. Add the account to the .procmailrc file on the machine where fetchmail is running
3. Set psequencer to connect to the Gmail IMAP server instead of your POP server

You can find how to setup panda.cfg for Gmail+IMAP in this section. For mail forwarding, you may need

```
define('SMART_HOST', 'your_smtp_server_name')dnl
```

in /etc/mail/sendmail.mc and then

```
$ make sendmail.cf
```

A recipe in ~/.procmailrc could be

```
:0 c
* ^From:.*atlpan@cern.ch
! Lelouch.Lamperouge@gmail.com
```

Mozilla Thunderbird

1. Select Tools | Accounts... from the menu
2. Click the desired email account
3. Select Server Settings
4. Click the "Leave messages on server" checkbox in the Server Setting section
5. Specify how long

Apple Mail

1. Click Mail and select Preferences...
2. Click the desired email account, then click the Advanced tab
3. Set Mail to remove copy from server after one day, after one week, ...

Contact Email Address: `hn-atlas-dist-analysis-help@cern.ch`

Responsible: Tadashi Maeno

-- TorreWenaus - 2016-04-24

This topic: PanDA > PandaSequencer
Topic revision: r1 - 2016-04-24 - TorreWenaus