Core summary

- Prepare method for Ganga applications
- Changes in the release tool and testing framework
- Unifying Ganga output

Mike Kenyon

Ivan Dzhunov

A prepare method for Ganga applications

- The purpose of the prepared state is to allow users to 'freeze' an application in a known state, such that the exact same analysis job can be executed in the future (optionally over different input data). During the prepare phase, files that are integral to the applications execution (such as custom binaries or experiment-specific software areas) are copied to the user's Shared Directory (ShareDir) which is by default <gangadir>/shared/<user>.
- This functionality already existed in the Athena backend, as exposed by the prepare() method, but the resulting configuration files were stored in the /tmp directory, the persistence of which cannot be predicted. Similar underlying techniques are now applied to Executable and Root applications, as well as to experiment specific ones.

Demonstration - Executable() application

Attribute attached to the application indicating whether it has been prepared

```
Out[23]: Executable (
 exe = 'echo' ,
 env = {} ,
 args = [Hello World] ,
 is_prepared = None
 )
```

The is_prepared attribute will hold a ShareDir object, which is generated with a random name in the users gangadir/shared directory:

```
Out[27]: ShareDir (
 name = '/home/mkenyon/gangadir/shared/mkenyon/conf-41540084-7372-1309946661-72' ,
 subdir = '.'
 )
```

As an example, we can configure the default Ganga job, which has the above Executable() application attached to it:

```
In [1]:a=Job()
In [2]:a.prepare()
Ganga.GPIDev.Schema                : INFO     Preparing Executable application.
Ganga.GPIDev.Schema                : INFO     Created shared directory: /home/gangadir/shared/conf-d7533df3-066a-4922-81ca-ec6404a10c6a
```

Note that it is equivalent to prepare the job, or the application associated with the job. In other words job.prepare() and job.application.prepare() are equivalent

Demonstration - Executable() application 2

submitting a job also automatically calls the prepare method behind-the-scenes. The result of running the prepare phase is a job with the following application attributes

```
Out[10]: Executable (
 exe = 'echo' ,
 env = {} ,
 args = [Hello World] ,
 is_prepared = ShareDir(name='/home/gangadir/shared/conf-d7533df3-066a-4922-81ca-ec6404a10c6a',subdir='.')
 )
```

The contents of the ShareDir depend on the type of application that was prepared. In the basic example above, the application would attempt to execute the command 'echo' on the backend/workernode, so we don't need to copy anything to the ShareDir. In a more realistic case, though, we might have a custom script that we wish to run on the worker node. This would then be copied to the ShareDir during the prepare phase.

Once an application has been prepared, it gains a reference counter which is stored in the Ganga metadata system, and can be checked by calling shareref:

```
In [10]:shareref
Out[10]:
                                  Shared directory |         Date created |  Reference count
 ------------------------------------------------- | -------------------- |  ---------------
                  conf-12754112-8042-1314361528-33 | 26 Aug 2011 13:25:28 |                 1
                     conf-86403407-38-1314361595-8 | 26 Aug 2011 13:26:35 |                 1
                  conf-24411345-7135-1314361709-42 | 26 Aug 2011 13:28:29 |                 1
                  conf-69852897-3565-1314541740-23 | 28 Aug 2011 15:29:00 |                 1
```

Demonstration - Executable() application 3

If an application is later associated with another job (j.copy()), or place in the box, the reference counter is incremented. Likewise, it's decremented when a job or box object is removed. ShareDirs with a reference count of 0 will be removed when Ganga next closes down. In the event that an application's ShareDir cannot be found during Ganga closedown, the application will be unprepared.

The contents of the shared directories can be viewed:

```
In [12]:shareref.ls('conf-284cbb3e-da37-4f6c-87e3-55576ec82cb7')
|  conf-284cbb3e-da37-4f6c-87e3-55576ec82cb7/
|  |  runMain.C
```

Applications not associated with a persisted Ganga object (such as a job or the box) can be prepared, but they, and their associated ShareDir, will not persist beyond the current Ganga session.

Copying a (prepared) application

Copying a prepared application/job object results in an identical copy of that object (i.e. referencing the same ShareDir) which will therefore have some of its attributes set read-only. It is possible that the user may wish to modify these attributes. This can be achieved by passing the unprepare argument to the copy method:

```
a=Job()
a.prepare()
b=a.copy(unprepare=True)
```

Note that by default (i.e. without the unprepare argument), calling copy() will not unprepare the application. It is possible to modify the default behaviour of the copy method such that an application will be unprepared when copied. This can be set either temporarily from the Ganga command line:

```
config['Preparable']['unprepare_on_copy']=True
```

or permanently, by adding the following to ~/.gangarc:

```
[Preparable]
unprepare_on_copy = True
```

Unpreparing an application

Once an application has been prepared, some of its attributes (as determined by that application's developers) will become read-only. This is to prevent the user accidentally changing an attribute on a prepared application (and hence violating the sense of a prepared application). In the event that the user really does want to modify a proteced attribute, they should either unprepare the application/job

```
j.app.unprepare()
app.unprepare()
```

or copy the application/job to a new instance in the following manner:

```
newjob=oldjob.copy(unprepare=True)
```

Applications can also be unprepared by resetting their is_prepared attribute to None:

```
In [1]:a.application.is_prepared=None
Ganga.GPIDev.Base.Proxy            : INFO     Unpreparing application.
```

Prepared Executable class

```
...
class Executable(IPrepareApp):
...
    _schema = Schema(Version(2,0), {
        'exe' : SimpleItem(preparable=1,defvalue='echo',typelist=
['str','Ganga.GPIDev.Lib.File.File.File'],comparable=1,doc='A path (string) or a File object specifying an executable.'),
    ...
    _exportmethods = ['prepare', 'unprepare']
    ...
    def unprepare(self, force=False):
        """
        Revert an Executable() application back to it's unprepared state.
        """
        logger.debug('Running unprepare in Executable app')
        if self.is_prepared is not None:
            self.decrementShareCounter(self.is_prepared.name)
            self.is_prepared = None
    ...
    def prepare(self,force=False):
        """
        A method to place the Executable application into a prepared state.
        The application will have a Shared Directory object created for it.
        If the application's 'exe' attribute references a File() object or
        is a string equivalent to the absolute path of a file, the file
        will be copied into the Shared Directory.
        When the application is submitted for execution, it is the contents of the
        Shared Directory that are shipped to the execution backend.
        """

        if (self.is_prepared is not None) and (force is not True):
            raise Exception('%s application has already been prepared. Use prepare(force=True) to prepare again.'%
(self._name))

        self.is_prepared = ShareDir()
        logger.info('Created shared directory: %s'%(self.is_prepared.name))

        #copy any 'preparable' objects into the shared directory
        send_to_sharedir = self.copyPreparables()
        #add the newly created shared directory into the metadata system if the app is associated with a persisted object
        self.checkPreparedHasParent(self)
        return 1
```

Changes in the testing framework

- Created another HTML page with top 25 tests (from all packages) that took longest time to execute
- made GangaPanda and GangaAtlas tests run in parallel , this way tests finish twice faster
- Fixed the IndexError at the end of step 5 of the release tool (running the tests), from which we used to recognize the tests are finished.

Changes in the release tool

- steps 3 (generation of config files) and 5 (running tests) - combining of all commands that needed to be executed (setting environment, update from SVN, running the actual command) in single script; release tool is printing the one line command that have to be executed on each of the accounts (ge, at, lb)
- automatic sending of emails to the Ganga email groups or to the release manager on different steps – pre-release ready, tests completed running, test reports ready, release ready;
    - release tool is asking before sending the emails because the release manager could be just playing/testing the tool
    - release manager doesn't need to modify (release version) and send the emails manually
- automatically clear some of the oldest pre-releases (if the number of pre-releases is above given threshold) to prevent running out of disk space when creating the new pre- and release
- document the changes on the twiki for the release procedure

Unifying Ganga output

- Separate presentation for it -> ...