Constructing a Testing Application for GWMS

LeRayah Neely-Brown, University of Illinois at Chicago – SIST Program | Bruno Coimbra – Fermilab

Introduction

Many of Fermilab's High Energy Physics experiments require High Throughput Computing to carry out simulations, data reconstruction, and data analysis. Glidein Workflow Management System (GWMS) is a tool that distributes High Throughput Computing. The purpose of GWMS is to provide convenient access to Grid resources and sites.

GWMS is coupled to HTCondor. HTCondor is the workload management system that is used for scheduling and job control. Users submit jobs to a local HTCondor queue and GWMS will make sure the job will run on one of the many remote resources.
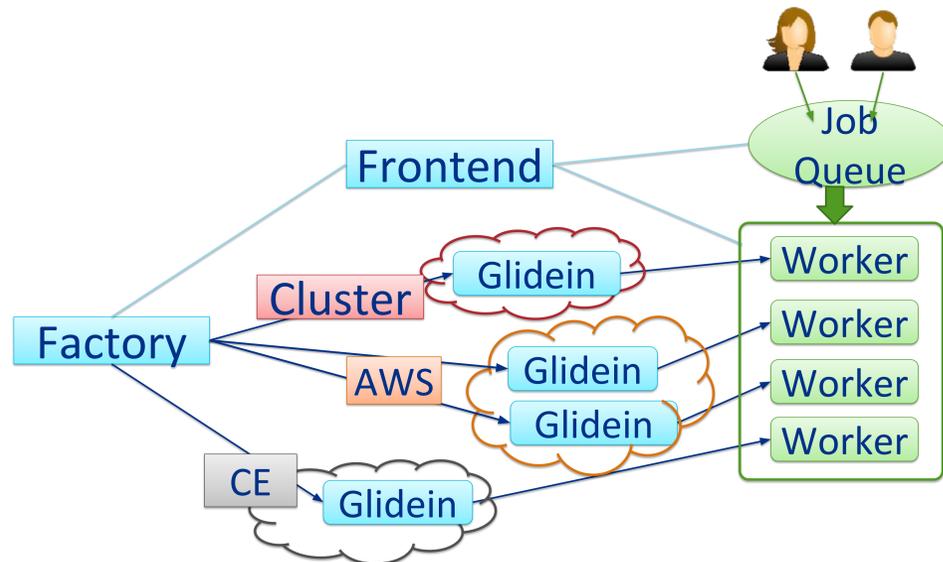

Figure 1 -The Operation of GWMS

Purpose for Testing Application

The GWMS Testing Application serves as a streamline tool that will directly submit a Glidein to the desired worker node. The testing application's goal to efficiently test the worker nodes of a site and test new GWMS configuration.

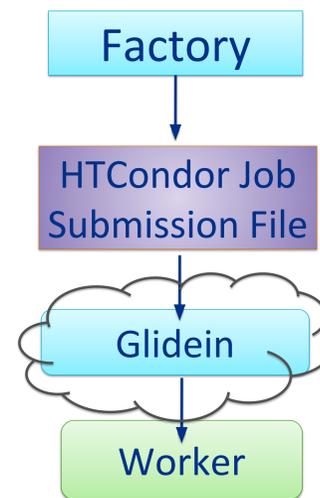## Method for Developing the Testing Application

Figure 2 - Workflow of Testing Application

The goal is to integrate the testing application with GWMS. The approach to develop the testing application is to start with existing GWMS scripts and improve the features to effectively send a Glidein with a modified configuration to the correct worker node. Studying the job submission file and the startup script file for the Glidein is essential to improve the process by making it more flexible.

```
# File: job.condor
#
Universe = grid
Grid_Resource = condor fermicloud089.fnal.gov fermicloud089.fnal.gov:9619
Executable = glidein_startup.sh
copy_to_spool = True
Arguments = $ENV(GLIDEIN_ARGUMENTS)
Transfer_Executable = True
transfer_Output_files =
WhenToTransferOutput  = ON_EXIT
stream_output = False
stream_error  = False
+TransferOutput = ""
x509userproxy = $ENV(X509_USER_PROXY)
+GlideinFactory = "$ENV(FACTORY_NAME)"
+GlideinName = "$ENV(GLIDEIN_NAME)"
+GlideinEntryName = "$ENV(GLIDEIN_ENTRY_NAME)"
+GlideinEntrySubmitFile = "$ENV(GLIDEIN_ENTRY_SUBMIT_FILE)"
+GlideinClient = "$ENV(GLIDEIN_CLIENT)"
+GlideinFrontendName = "$ENV(GLIDEIN_FRONTEND_NAME)"
+GlideinCredentialIdentifier = "$ENV(GLIDEIN_CREDENTIAL_ID)"
+GlideinSecurityClass = "$ENV(GLIDEIN_SEC_CLASS)"
+GlideinWebBase = "$ENV(GLIDEIN_WEB_URL)"
+GlideinLogNr = "$ENV(GLIDEIN_LOGNR)"
+GlideinWorkDir = "$ENV(GLIDEIN_STARTUP_DIR)"
+GlideinSlotsLayout = "$ENV(GLIDEIN_SLOTS_LAYOUT)"
+GlideinMaxWalltime = $ENV(GLIDEIN_MAX_WALLTIME)
+fename = "$ENV(GLIDEIN_USER)"
```

Figure 3 - Example of a Job Submission File Before Modifications

Technologies Used

The technologies used to construct the testing application were Python 3, Shell script, FermiCloud (OpenNebula), and Git. These technologies are essential because they are easily accessible and available for Windows or Linux/Unix users that will utilize GWMS.

Challenges + Future Steps

Some challenges faced while creating the testing application were modifying the startup script for the Glidein to help format the testing application. The route of editing and adding new code to the script would cause some Glideins to fail to reach the correct worker node or be idle.

For future work, a web framework will be created to visualize the result of the testing application and allow remote manual submission of the Glidiein. Also, a feature to override some parameters received from the Factory and Frontend of GWMS for the Glidein will be implemented.

Acknowledgements

I would like to thank my mentor Bruno Coimbra, the leader of the GWMS project, Marco Mambelli, and all of the developers who work on the GWMS project. I would also like to thank Fermilab, the SIST Committee, and SIST Mentors for giving me the opportunity to partake in this project.