Fermilab U.S. DEPARTMENT OF ENERGY | Office of Science

HEPCloud Stakeholder meeting

Moving Frontend functionality to HEPCloud

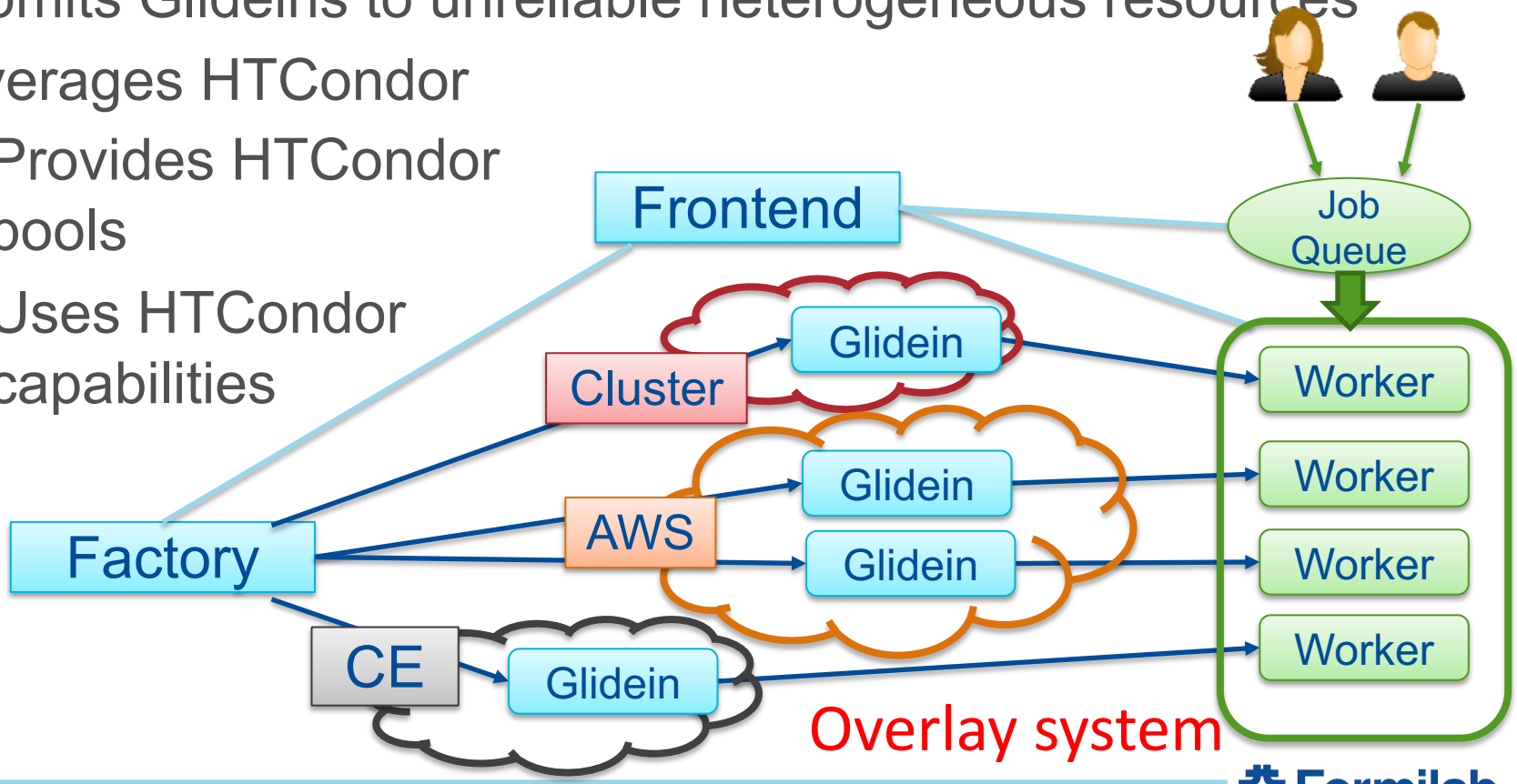
Marco Mambelli

12 August 2020

GlideinWMS

GlideinWMS is a pilot based resource provisioning tool for distributed High Throughput Computing

- Provides reliable and uniform virtual clusters
- Submits Glideins to unreliable heterogeneous resources
- Leverages HTCondor
  - Provides HTCondor pools
  - Uses HTCondor capabilities



Fermilab

Glidein: node testing and customization

- Scouts for resources and validates the Worker node
  - Cores, memory, disk, GPU, …
  - OS, software installed
  - CVMFS
  - VO specific tests
- Customizes the Worker node
  - Environment, GPU libraries, …
  - Starting containers (Singularity, …)
  - VO specific setup
- Provides a reliable and customized execute node to HTCondor

Factory

- A Glidein Factory knows how to submit to sites
  - Sites are described in a local configuration
  - Only trusted and tested sites are included
- Each site entry in the configuration contains
  - Contact info (hostname, resource type, queue name)
  - Site configuration (startup dir, OS type, …)
  - VOs authorized/supported
  - Other attributes (Site name, core count, max memory, ...)
  - Glideins can also auto-detect resources
- Configuration can be auto-generated (e.g. from CRIC), admin curated, stored in VCS (e.g. GitHub)
- Condor does the heavy lifting of submissions.

Frontend

- Monitors jobs to see how many Glideins are needed
- Compares what entries (sites) are available
- Requests Glideins from the Factory
- Requests Factory to kill Glideins if there are too many
- Pressure-based system
  - Works keeping a certain number of Glideins running or idle at the sites
  - Glideins requests are gradual to avoid spikes and overloads
- Manages credentials and delegates them to the Factory.

Proposal

- Will have to be coordinated and agreed by both projects' stakeholders

- Migrate Frontend functionalities to HEPCloud
- Phase out GlideinWMS Frontend
- Current GlideinWMS users will use HEPCloud software

Plan – HEPCloud software

- Migration to Python3 of GlideinWMS code
- Write a HEPCloud Channel to provide Frontend functionalities
  - Modular python3 code reproducing Frontend decisions starting with similar constraints
  - Code integrated in the DecisionEngine (configuration, logging, monitoring)
- Separate out the modules to communicate w/ the Factory to make them available to multiple channels
- Separate out clearly the Glidein functionality to make it available to multiple channels
  - Testing and setup
  - Pilot only for channels using that

Plan – GlideinWMS Frontend phase out

- Tentative, to be discussed with and agreed by GlideinWMS Stakeholders
- Communication w/ Stakeholders, agreement at the next meeting (9/9)
- Freeze of GWMS Frontend features
- Timeline for bug fixes and security updates
- Discussion of HEPCloud software requirements
- Timeline for HEPCloud software availability

Plan - HEPCloud software expected requirements

- Compatible features
- Documented load
- Simple deployment
  - Packaging, e.g. RPMs distributed in yum repo
  - Installation and operation instructions
  - Base configuration
  - Running in containers