HEPCloud Resource Provisioning

Anthony Tiradani

OSG Blueprint Meeting

21 February 2018

HEPCloud Target Resources

- Fermilab HEPCloud instance currently supports provisioning compute resources.
  - Delivered in the form of a glideinWMS pilot
  - Uses the glideinWMS Factory for provisioning
  - Architecture allows for the use of other provisioners
- HEPCloud can provision resources from Cloud Providers, OSG sites, and HPC sites.
  - On-going effort to use HTCondor-CE as an submission point into some HPC sites
  - Currently using the HTCondor SSH interface to submit to NERSC
- HEPCloud future work:
  - Provisioning storage and data movement infrastructures
  - Provisioning services other than batch computing related services (talking with VC3 project)

HEPCloud Allocation Models

- HEPCloud will be able use dedicated, opportunistic, pay-per-use, and allocation-based provisioning models.
- **Dedicated** resources are resources that already exist, therefore do not need to be provisioned. These resources are treated as preferred resources. All other resources are considered expansion resources.
- **Opportunistic** models are handled similarly to the glideinWMS model
- **Pay-per-use** models go through an cost optimization algorithm to select the most cost effective range of resources that meet the workflow requirements
- **Allocation** based models are treated as a simplified pay-per-use algorithm
- HEPCloud is developing an **economic model** in an attempt to do apples-to-apples comparisons of the “cost” of computing between the different models.

HEPCloud Resource Allocation

- The Fermilab instance of HEPCloud makes use of the glideinWMS
- As noted in the glideinWMS talk, glideinWMS does not allocate resources, it generates resource requests
- Allocation of resources to jobs is done by HTCondor
- HEPCloud keeps track of its use of:
  - Allocations similar to that on HPCs
  - Budgeting and finances for Clouds
  - This requires code and policy configuration (on-going task targeted for late 2018)
- Resource request distribution is based on job requirements, resource provider capabilities, estimated costs, budgets, etc.
- Resource locations are manually added at the moment
  - Cloud resources require on-demand infrastructure, e.g. CVMFS
  - Each and every HPC is unique, requiring vetting and curation

HEPCloud Resource Allocation (cont.)

- The HEPCloud Architecture allows for multiple provisioners.
  - Have looked at rvgahp[1] for pull models (glideinWMS project is currently looking at it as well)
  - glideinWMS is primarily a push model provisioner
- HEPCloud provisions based on needs.
  - Currently, single core, whole node, multi-node pilots are in use

[1] https://github.com/juve/rvgahp

HEPCloud Decision Engine

- HEPCloud is developing the Decision Engine (DE).
  - The DE Framework allows for deterministic, reproducible, decision making workflows called Decision Channels
  - Decision Channels make up the logic by which decisions are made to request (or not) resources from a provider
  - Decision Channels may depend on other Decision Channels
  - Decision Channels are made up of contributed modules and configured “business rules”
- Decision Channel modules and configurations are being developed for the Fermilab instance
- Once Fermilab has more experience operating the DE, a set of reference modules and configurations will be released

Runtime Content

- HEPCloud uses the glideinWMS pilot to set up the runtime environment
  - For cloud resources, the glideinWMS project created a pseudo-service that reads required information from the “user-data” and bootstraps the pilot
  - For HPC resources, HEPCloud is using glideins to provision multiple machines
  - All resources report back to the HEPCloud pool
- HEPCloud anticipates being able to set environments through the use of VMS (Cloud) and containers (HPC, Cloud if deemed useful).

Differences w.r.t. glideinWMS

- glideinWMS provisioning algorithm optimized for the traditional grid model
- HEPCloud allows for BYOA (Bring Your Own Algorithm)
  - Facility/instance driven
  - Capabilities exposed only to instance admins
  - Integrates various monitoring sources as part of the decision making process
- HEPCloud architecture can expand beyond batch computing
  - Long term plans include
    - investigating data movement
    - provisioning other computing environments
  - Can expand to use multiple types of provisioners

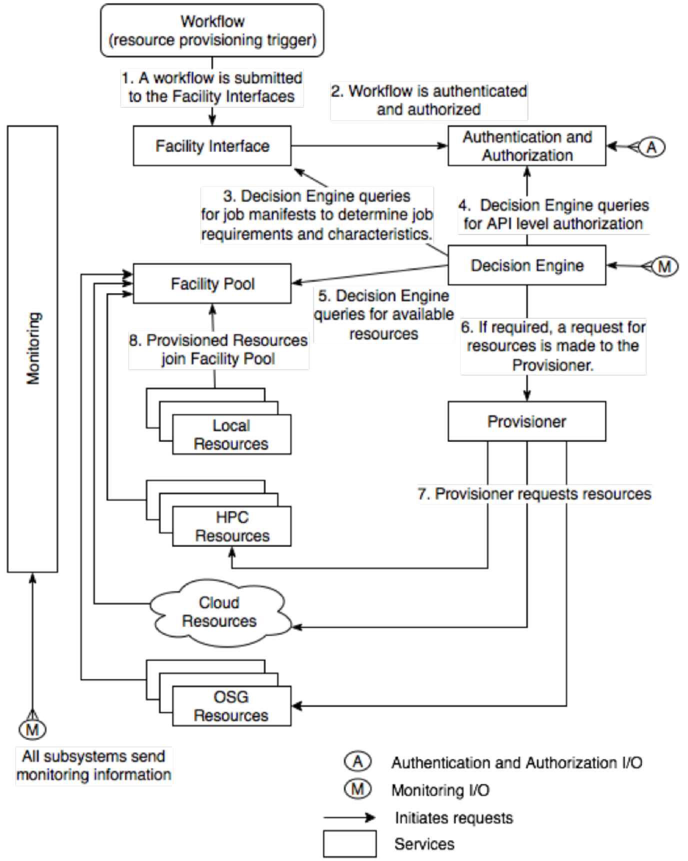
Requirements for Resource Providers

- Web cache available
  - Absence can be worked around (similar to what we do at NERSC)
    - Work arounds not sustainable or desirable
  - Have to build your own infrastructure in cloud, but it is available
- Provide container capability (e.g. Singularity, Shifter, etc.)
  - Would be nice if sites would standardize on a solution and configuration
    - For example, NERSC only allows Shifter, almost all other HPC sites HEPCloud has looked at only allow Singularity
    - The Singularity configurations are not yet standardized at sites
- The Fermilab instance of HEPCloud uses the glideinWMS factory to provision
  - This means that HEPCloud can support whatever submission endpoint HTCondor supports

Backup Slides

Fermilab HEPCloud Instance Architecture

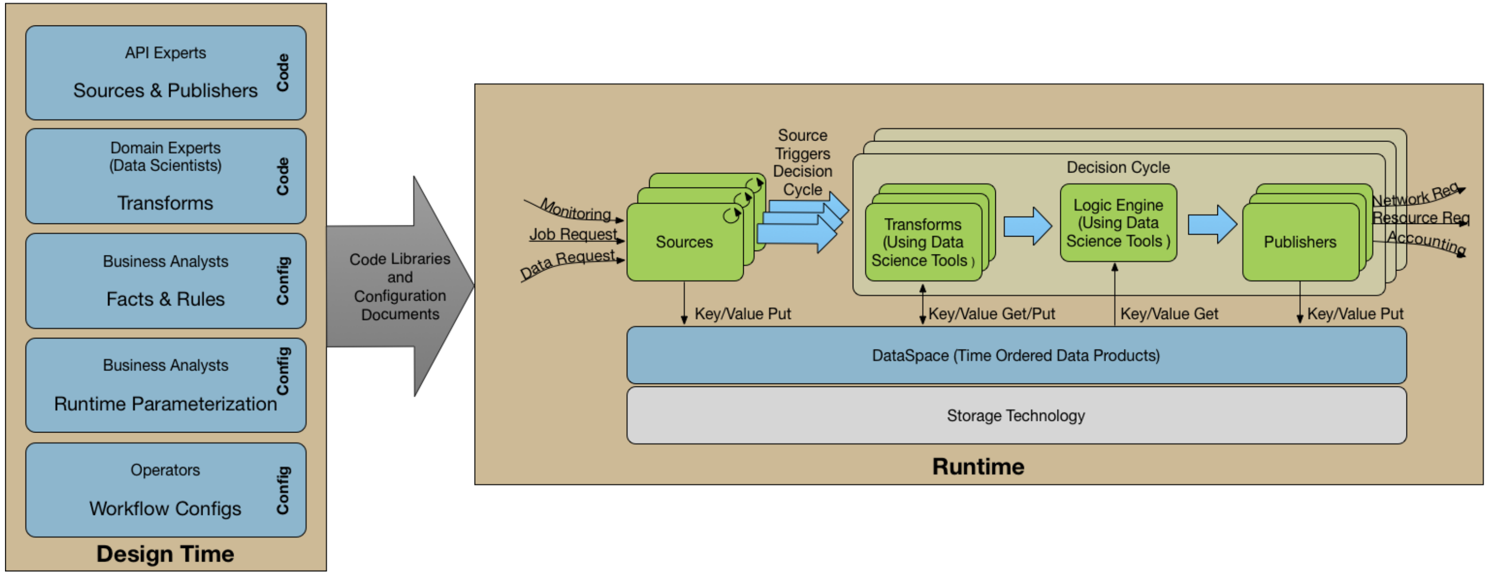
- Specific to Fermilab
- Concentrates on expanding batch computing
- New component: Decision Engine (DE)
- Integrates Monitoring as a feedback loop informing decisions
- Plan for production status in late 2018



Workflow (resource provisioning trigger)
1. A workflow is submitted to the Facility Interfaces
2. Workflow is authenticated and authorized
Facility Interface
Authentication and Authorization
A
3. Decision Engine queries for job manifests to determine job requirements and characteristics.
4. Decision Engine queries for API level authorization
Monitoring
Facility Pool
Decision Engine
M
5. Decision Engine queries for available resources
6. If required, a request for resources is made to the Provisioner.
8. Provisioned Resources join Facility Pool
Local Resources
Provisioner
7. Provisioner requests resources
HPC Resources
Cloud Resources
OSG Resources
M
All subsystems send monitoring information
A Authentication and Authorization I/O
M Monitoring I/O
Initiates requests
Services

HEPCloud Decision Engine (DE) Architecture

- In the Fermilab context, replaces the glideinWMS Frontend
- Plan for production status in late 2018



Fermilab