EOSC Technical Specification

Hub Federated Core Services

HTC/HPC Compute - Multitenant job submission

Introduction

Most scientific challenges require running computationally demanding tasks. Typically, these computing challenges can be tackled by gathering several computing resources that concurrently run the tasks. In some cases, the computational problem can be addressed by multiple and loosely coupled tasks that can run over different data blocks or different parameter sets, and in some cases, the problem requires gathering several computing elements together to solve every single task in a closely coupled parallelism. The former is addressed through High-Throughput Computing (HTC) execution approaches and the latter by the High-Performance Computing (HPC) ones. In the HTC/HPC Compute TCOM we address services for running a large set of independent tasks and to jointly use several computing resources to run a parallel job.

In this specification, we expose a macro-feature for Multitenant job submission, which relates to the capability of submitting HPC/HTC jobs with predefined constraints (both at resources and software) without a previously deployed virtual infrastructure. This service should be able to run a bunch of batch job on HTC/HPC and cloud compute resources, interfacing with storage solutions and seamlessly integrated with the authentication mechanisms. The main difference between HPC and HTC jobs will be the requirement of multiprocessing (OpenMP or MPI for example). Some sites support both types of jobs, using different queues and specifications in the batch job to differentiate and provision the rightmost resources.

Adopted Standards

Standard / Specifications	Short description	References
DIRAC4EGI WMS Rest API	A central service to submit job specifications to a variety of backends, including the management of data.	https://dirac.readthedocs.io/en/latest/
OpenStack (especially Nova API and Keystone API v3)	OpenStack is an Open Source cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter, all managed and provisioned through APIs with common authentication mechanisms.	OpenStack API
OpenID tokens & SAML	The service must leverage habitual authentication and authorization mechanisms used in Scientific Research Infrastructures to avoid users to handle multiple credentials. EGI Check-in is an example (https://wiki.egi.eu/wiki/AAI_guide_for_SPs)	https://openid.net/connect/, http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.pdf
CREAM	The CREAM (Computing Resource Execution And Management) Service is a simple, lightweight service that implements all the operations at the Computing Element level	https://cream-guide.readthedocs.io/en/latest/Overview.html
DRMAA	The 'Distributed Resource Management Application API (DRMAA)' is an API specification for tightly coupled and portable programmatic access to cluster, grid, and cloud systems.	https://www.drmaa.org/

High-level Service Architecture

The service will require multi-tenant platform services to deal with different users. Figure 1 describes the Core Services and the HTC and cloud endpoints.

A duly authorised user (e.g. by using group attributes on the credential, issued by a trusted User Membership service) should be able to submit a job through a Job Management client (which could be a REST client) to a Workload Management Service (WMS), which will provide temporary and limited local storage for the input/output of the jobs until they are executed and output is retrieved. According to the job specification and the infrastructure information system, the WMS service will decide where to run the jobs, which could imply an existing HTC/HPC execution endpoint or spawning a job agent on a worker Virtual Machine in the cloud.

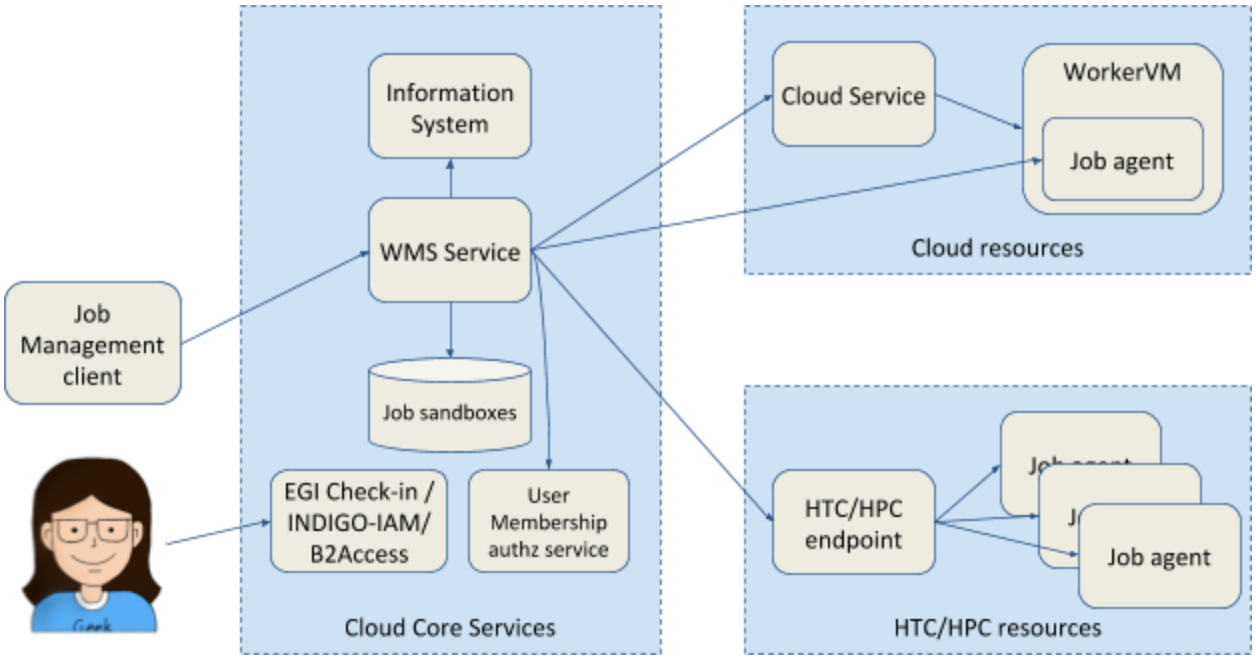

Figure 1: Potential high-level architecture for the Multitenant job submission Macro-feature.

The actual resources are managed by Local Resource Management Systems and Cloud IaaS/Paas at the infrastructure level.

Interoperability guidelines

With respect to the user, the service should expose the following interface:

- Workload management - at least: submit a new job, list the jobs submitted by a user, get the status of a single or several jobs, get detailed information of a job (such as the specification, details on the credentials, history of statuses the job has gone through, allocated resource information, etc), kill a running job, get the output of a finished job (desirably also from a running job), delete a finished (or several) job.

- Identity management and delegation - at least: authenticate in a provider to retrieve a credential (token, certificate, etc., potentially through the service) valid for a specific period, get credential information, invalidate credential.

Therefore, we expect the service to interact with the following services:

AAI interoperability

- Services should provide access to users authenticated with one of the EOSC-hub AAI federated identity protocols (OpenID Connect and/or SAML).
- Potentially Token Translation services to associate temporary proxies to identities.

Orchestration interoperability

- Services may make use of the API of the cloud Management services to deal with the cloud backends.

High-Throughput Compute endpoints

- The WMS will interact with the site-level HTC endpoints for the managing of jobs.

Federation-level information services

- The WMS will interact with the information system of the federation for the resource matchmaking.

Examples of solutions implementing this specification

- DIRAC4EGI: https://dirac.egi.eu/DIRAC/
- INDIGO-C Orchestrator (https://github.com/indigo-dc/orchestrator) + QCG-Computing (http://www.qoscosgrid.org/trac/qcg-computing)
- CREAM (Computing Resource Execution and Management) - https://cream-guide.readthedocs.io/en/latest/Overview.html