Dirac integration with a general purpose bookkeeping DB: a complete general suite for distributed resources exploitation

M Chrzszcz, C De Santis, G Donvito, A Fella, R Grzymkowski, B Santeramo, L Tommassetti, M Zdyba

E-mail: bruno.santeramo@ba.infn.it

Abstract. In the context of High Energy Physics computing field the R&D studies aimed to the definition of the data and workload models have been carried on and completed by the Super$B$ community beyond the experiment life itself. The work resulted of great interest for a generic mid- and small size VO to fulfill Grid exploiting requirements involving CPU-intensive tasks.

We present the R&D line achievements in the design, developments and test of a distributed resource exploitation suite based on DIRAC. The main components of such a suite are the information system, the job wrapper and the new generation DIRAC framework. The DB schema and the SQL logic have been designed to be able to be adaptive with respect to the VO requirements in terms of physics application, job environment and bookkeeping parameters. A deep and flexible integration with DIRAC features has been obtained using SQLAlchemy technology allowing mapping and interaction with the information system. A new DIRAC extension has been developed to include this functionality along with a new set of DIRAC portal interfaces aimed to the job, distributed resources, and metadata management. The results of the first functionality and efficiency tests will be reported.

1. Introduction

2. Description of suite design

philosophy: simple, standard and long term solution
bird's eye view all over the project

3. The Dirac extension

- extension structure description
- Dirac general purpose project short description + Dirac configuration
- service and systems description in detail
- web interface components description

4. Bookkeeping DB integration

- Software layer based on SQLAlchemy
– advantages using SQLAlchemy: Object relational Mapping, clean code, fast

– development, change of DB backend
- BK description, highlighting the general purpose characteristics
– session, request, dataset concept

5. Job wrapper component

- general workflow
- data management policy: stage-in and stage out strategies

6. Simulation production use case: the SuperB experience

- general description: workflow, Dirac portal design
- past experience: the webui project
– Session definition interface –¿ DB dynamic build up

7. Test session

A test session was performed aiming to demonstrate the awareness of SuperBDIRAC to integrate the monitoring of a bookkeeping database in DIRAC.

7.1. Goal description

DIRAC capabilities and performance as job management tool are well documented (insert some reference). Super$B$ distributed simulation stack (WebUI + SBK + Severus) was successfully used in 3 simulation campaigns (ask to Armando for confirmation and references).
Test goal was to demonstrate SuperBDIRAC is capable of substitute WebUI as monitoring tool for job submission, showing data from a bookkeeping database. Test ”Q-factor” was the exact corrispondence of information as stored in SBK and displayed in SuperBDIRAC, like in WebUI monitoring page. Correct execution of simulation jobs is not important while the bookkeping information are properly managed.

7.2. Testbed description

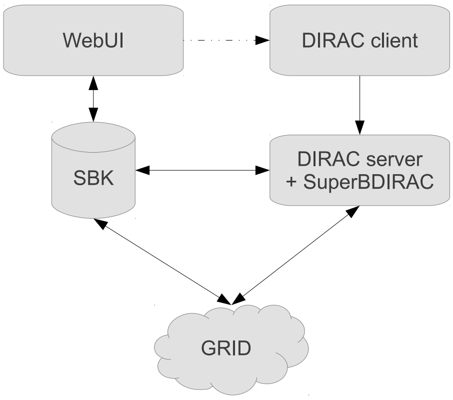

Figure 1. Testbed schema.

FastSim job submission is created via WebUI interface (see figure to be inserted). Once submission is created, a set of scripts and configuration files are created whose path is reported by WebUI interface: in particular the path of php submission script is taken to be be parsed. Since WebUI is still linked with SBK, this portal acts as well as monitoring portal, useful for a check-cross between info displayed in SBK, WebUI and SuperBDIRAC.

A python script (mc_production.py) acts as parser for php submission script and interface with DIRAC client. Once taken all submission parameter, mc_production.py uses DIRAC API to submit jobs to DIRAC server.

DIRAC server receive jobs from DIRAC client, than starts the normal workflow for job management in DIRAC: scheduling, pilot submission, payload retrieval and execution, stageout. Since DIRAC server is equipped with SuperBDIRAC, even bookkeeping monitoring is performed by this component.

In SBK the job submission insertion is performed by WebUI, while later info are updated by Severus script executed with every job submitted. Severus interact with SBK via REST interface, while WebUI and SuperBDIRAC have direct acces to SBK since they are in the same LAN area (ask to Armando for confirmation).

Jobs were submitted via grid by DIRAC server. Submission was performed using WMS instead of using direct submission to CREAM CE.

Submission site was INFN-T1, which ensured CPU time to execute latest simulations related to Super$B$ once the experiment closure.

7.3. Test description

Every job simulated 3000 events: this value was chosen to have an execution time of about 10 minutes. Physical parameters were the same of several official productions. 3 main bunch submission of 400 jobs were performed at INFN-T1 to obtain a total 1200 simulations. In addition, 2 bunch submission of 10 jobs were performed, again at INFN-T1, to force some failure message in SBK and catch it even in SuperBDIRAC monitoring. In summary, 1220 jobs were submitted for test.

Since test goal, several screenshots of WbUI and SuperBDIRAC monitoring were saved, in addition to SBK data screenshots.

7.4. Results and conclusions

8. Conclusions

- We are offering a Dirac extended suite capable to satisfy the needs onsmall and mid size VOs in terms of distributed resource exploitation....

References

[1] Fielding R T 2000 *Architectural Styles and The Design of Network-based Software Architectures* , PhD Thesis, University of California Irvine

[2] A.Fella, E.Luppi, L.Tomassetti *A General Purpose Suite for Job Management, Bookkeeping and Grid Submission.* International Journal of Grid Computing & Applications (IJGCA) Vol.2, No.2, June 2011. DOI: 10.5121/ijgca.2011.2202.