DIRAC Production Manager Tools

Presenter: Gennady Kuznetsov
for the LHCb collaboration
• Introduction to DIRAC

• DIRAC Authoring Tools

• Production Repository and Production Agent

• Automatic Data Processing

• Conclusion
DIRAC – Distributed Infrastructure with Remote Agent Control

- **LHCb Workload and Data Management system** for the distributed data production and physics analysis. (Monte-Carlo simulation, reconstruction…)
- Composed of a set of light-weight services and a network of *distributed agents* to deliver workload to computing resources.
- Integrates computing resources available at LHCb production sites as well as on the LCG grid.
- It was demonstrated to support 5000 simultaneous jobs across 65 sites with no observed limitation.
- Currently handles over 70 TB of data.

(See talk by Dr. TSAREGORODTSEV, Andrei [301]-DIRAC, the LHCb Data Production and Distributed Analysis system)
The Production Manager plays a major role in executing organised data production activities within LHCb on a very large scale. His main tasks are:

• Designing data processing sequences
• Creating many thousands of data processing jobs
• Submitting and monitoring jobs
• Verifying output files
• Replicating data files in different centres

All of these activities can be efficiently fulfilled only by using automated tools to assist the Production Manager in handling many thousands of distributed jobs and files.
The *Job* performs a certain *activity* that can be described by a complex succession of operations (e.g. running applications).

- Run Monte Carlo Simulation and write a file `xxxxx_1.sim` and create spill over events in `xxxxx_2.sim` and `xxxxx_3.sim`
- Run Boole digitization program, read detector simulation and spillover files and reconstruct detector response output in `xxxxx_4.digi`
- Run Brunel reconstruction program, read digitization data and reconstruct `xxxxx_5.dst`

Each operation takes input data, produces output data and defined by parameters.

The *Workflow* in this case is an abstract description of the sequence of operations to be performed.

The *Workflow Template* is derived by specifying the internal parameters of *Workflow*

- *Gauss v15r5*, *LHCb v15r4*, *XMLDDDB v22r0*, *Boole v5r3*, *Brunel v23r5* (i.e. specifying version)
- Specifying event type = 61, Number of events = 500
Each **Workflow** can be decomposed into a series of **Steps** connected to each other via input-output files.

A **Step** is the smallest entity that could run independently as a job (if persistent storage is used for the input-output files).
Step usually consists of a series of smaller unbreakable operations - not necessarily producing output nor requiring input files, but may produce, exchange or share some data.

To brand these operations as reusable components, we introduce the smallest entity called a Module.

Instances of the Modules can be configured or linked with each other by defining their parameters.
Authoring Tools

- Module Editor
- Step Editor
- Workflow Editor

XML file

Workflow Library

Presenter Name: Gennady Kuznetsov
The *Production Repository* is a hierarchical storage designed to provide an area for the Jobs handling. It is populated by a *Production Manager Console* and its content is used by a *DIRAC Production Agent*.

**Production Repository**

- **Repository**
  - Production Templates
  - Productions
  - Jobs

**Production Manager**

- **Workflow Library**
  - **Production Manager Console**
    - **Command line tools**
      - Submits Jobs; Checks Job Status; Downloads output sandbox;
      - **Production Repository**
        - MCPProduction template
          - MCPProduction1
          - MCPProduction2
          - Job 1
            - ......
            - Job n
          - MCPProduction3
          - Reconstruction template
            - Reconstruction1

**Production Agent**

- **Dirac WMS**
Submitting production jobs

Creates and publishes Production

Repository

Production Template

Production

Job

Production Agent

Template + Parameters

Workflow + Parameters

Production Manager

DIRAC WMS

DIRAC

Computing Resource
Data processing is automated through a *Processing Database* and a series of *Transformation Agents*.

The appropriate *Transformation Agents* are triggered by the presence of a sufficient number of file entries in the *Processing Database*. The *Transformation Agent* creates the required number of jobs in the *Production Repository* which are then picked up by the *Production Agent*. 
Automatic job submission

Creates and publishes Transformation

Associates with the Production

Transformation Agent

Processing DB
Transformations

File Records

Repository
Production Templates

Production
Jobs

Production
Agent

Production Console

Production Manager

DIRAC WMS

DIRAC

Computing Resource

Presenter Name: Gennady Kuznetsov
Automatic job submission

- Repository
  - Production Templates
- Production
- Job
- Production Agent

Production Console

File entry

Production Manager

Transformation Agent

Processing DB
  - Transformations
  - File Records

DIRAC WMS

DIRAC

File entry

Resource
Operational experience in DC04

• The Production Console was used during the DC04 production period
  – more than 100,000 jobs were created and executed
  – 10 Workflows of 5 to 12 application Steps were used
  – The whole production was handled by a small team of production managers with just one person on shift at each moment
Use cases in DC06

• In the DC06 LHCb will test its complete Computing Model which includes
  – Data acquisition at CERN with subsequent inline replication to Tier1 centres
  – Automatic submission of reconstruction and filtering jobs at Tier1 centres as soon as data become available
  – DST data replication across the Tier1 centres

• The Production Manager tools will be used to cover all these new tasks to minimize the human intervention and to increase responsiveness of the production system
Conclusion

- The DIRAC Production Manager tool suite allows to define Workflows of arbitrary complexity
- It allows to handle seamlessly large numbers of jobs by a single Production Manager
- The automatic job preparation and submission triggered by the input data availability reduces the need for the human intervention
- The Production Console was successfully used in the LHCb DC’04.
- The complete system to be used in DC06 will support all the production operations defined in the LHCb Computing Model
Backup Slides
Variables

How we can avoid variable clash if different programmers are creating Modules and Steps? If we use multiple instances of the same Module – we shall have a local scope.

<table>
<thead>
<tr>
<th>Option Name: var1</th>
<th>Option Name: var2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Type</td>
</tr>
<tr>
<td>Value</td>
<td>Value</td>
</tr>
<tr>
<td>Linked Instance</td>
<td>Linked Instance</td>
</tr>
<tr>
<td>Linked Option</td>
<td>Linked Option</td>
</tr>
<tr>
<td>Input Flag</td>
<td>Input Flag</td>
</tr>
<tr>
<td>Output Flag</td>
<td>Output Flag</td>
</tr>
<tr>
<td>Description</td>
<td>Description</td>
</tr>
</tbody>
</table>

Options grouped in a form of list.
All our objects (Module, Step, Workflow) are lists of Options.

Global scope
$VAR2 = “string”
var3 = $VAR2
var1 = $VAR2

Name space based scope
var3 = inst1.var2
var1 = self.var2

Gennady Kuznetsov
class HistogramTest:
    seed = 3
    numbers = 200
    filename = "WorkflowHistogramTest1.155"
    def execute(self):
        ins1 = M1()
        <skipped>
        ins1.execute()

        ins2 = M2()
        ins2.seed = self.seed
        ins2.filename = 'default_file_name'
        ins2.mu = inst1.mu
        ins2.sigma = 2.5
        ins2.numbers = self.numbers
        ins2.execute()
DIRAC Production Manager tool includes

• flexible and *experiment (content) independent* framework to formulate complex multi-staged jobs

• *LHCb specific* Workflow library combined from the smaller and reusable components

• tools for the bulk Job preparation, submission and control based on *Production Repository* and *Production Agent*

• automated data processing tools based on *Processing Data Base* and *Transformation Agents*