# DIRAC Infrastructure For Distributed Analysis

```
from DIRAC.Client.Dirac import *
dirac = Dirac()
job = Job()
job.setApplication('DaVinci', 'v12r11')
job.setInputSandbox(['Application_DaVinci_v12r11/DV_Pi0Calibr.opts', 'Application_DaVinci_v12r11/lib '])
job.setInputData(['/lhcb/production/DC04/v2/DST/00000746_00002032_9.dst','/lhcb/production/DC04/v2/DST/00000746_00002038_9.dst '])
job.setOutputSandbox(['pi0calibr.hbook', 'pi0histos.hbook'])
jobid = dirac.submit(job,verbose=1)
print "Job ID = ",jobid
```
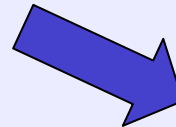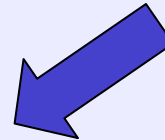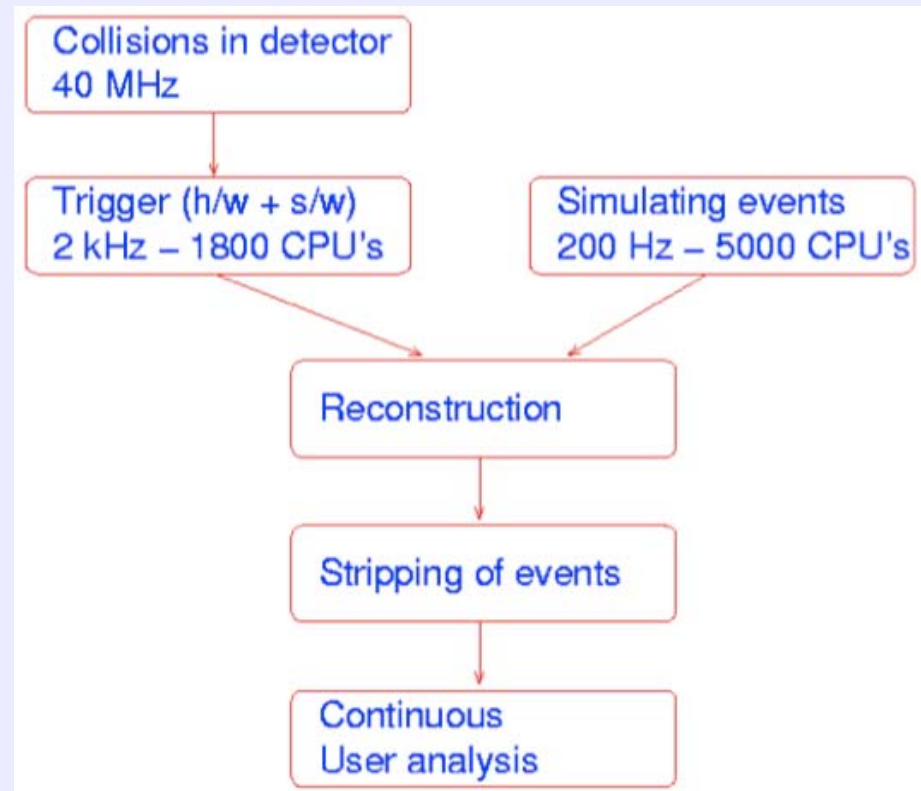
IoP Particle Physics 2006 Warwick
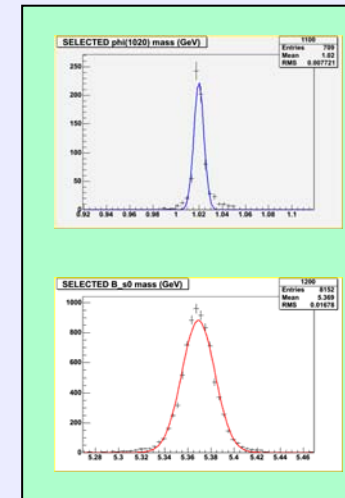
- **Distributed Analysis in LHCb**
- **Introduction to DIRAC**
  - DIRAC API
  - Workload Management System
- **WMS Strategies & Performance**
- **Outlook**

Distributed Analysis in LHCb

- **Stripping is centrally managed analysis run as a production**
  - Reduces analysis datasets to a size of 10^6 to 10^7 events per year
- **The user analysis is performed on stripped data throughout the year**
  - User analysis will take place at all LHCb Tier 1 centres
  - All stripped data will be disk resident and replicated to all sites
- **No direct submission to LCG for LHCb Analysis jobs**
  - Instead submit via DIRAC

Collisions in detector
40 MHz

Trigger (h/w + s/w)
2 kHz – 1800 CPU's

Simulating events
200 Hz – 5000 CPU's

Reconstruction

Stripping of events

Continuous
User analysis

- The DIRAC Workload & Data Management System (WMS) is made up of Central Services and Distributed Agents
  - Realizes PULL scheduling paradigm
    - Agents are requesting jobs whenever the corresponding resource is available
    - Execution environment is checked before job is delivered to WN
  - Service Oriented Architecture masks underlying complexity
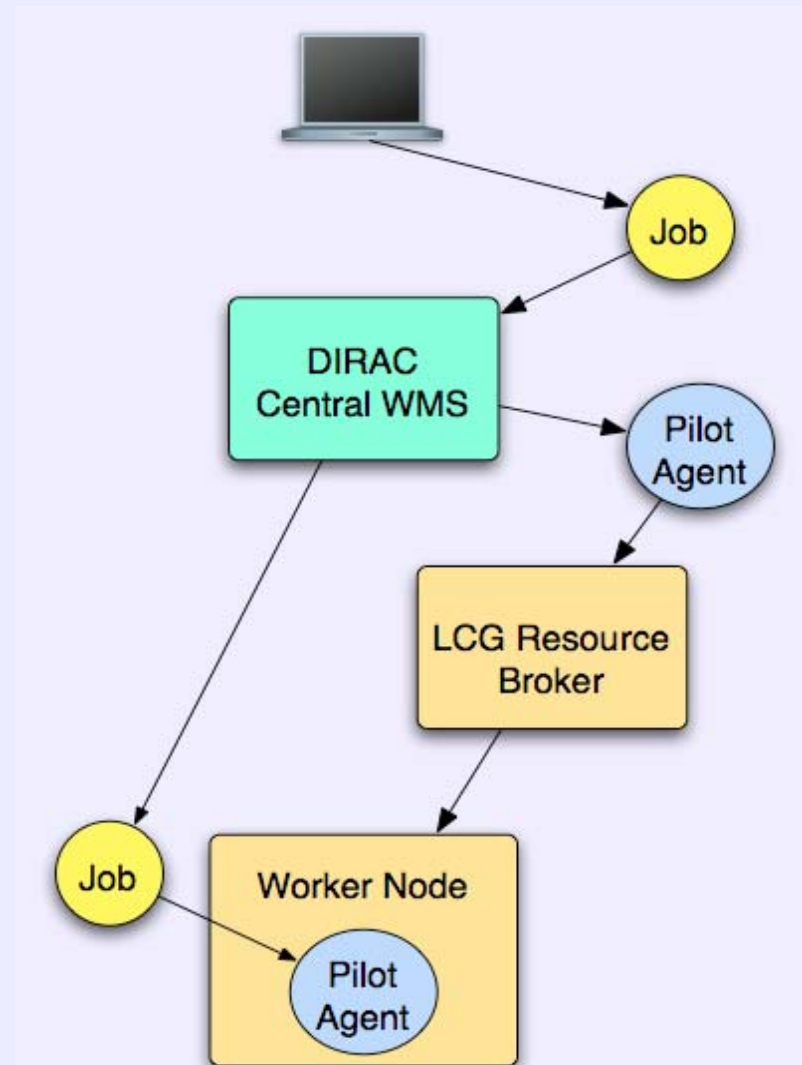
The DIRAC API

- The DIRAC API provides a transparent way for users to submit production or analysis jobs to LCG
  - Can be single application or complicated DAGs
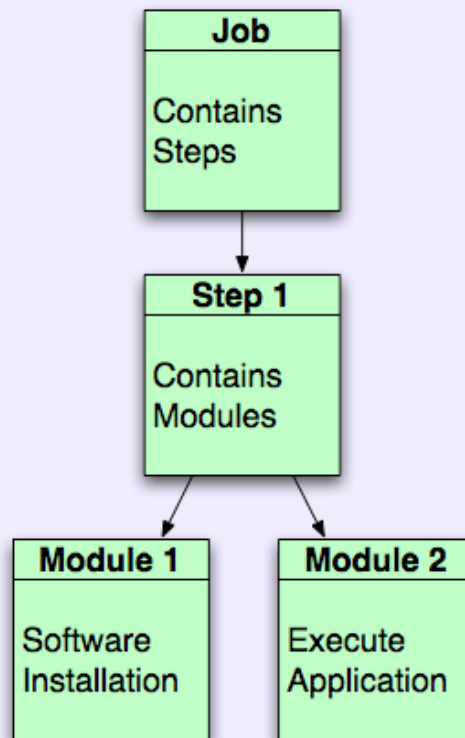

```
from DIRAC.Client.Dirac import *
dirac = Dirac()
job = Job()
job.setApplication('DaVinci', 'v12r15')
job.setInputSandbox(['DaVinci.opts', 'lib '])
job.setInputData(['/lhcb/production/DC04/v2/DST/
00000742_00003493_10.dst'])
job.setOutputSandbox(['DVNtuples.root', 'DaVinci_v12r15.log'])
jobid = dirac.submit(job,verbose=1)
print "Job ID = ",jobid
```

- While it may be exploited directly, the DIRAC API also serves as the interface for the GANGA Grid front-end to perform distributed user analysis for LHCb

LHCb DIRAC Monitoring

LHCb DIRAC Monitoring

DIRAC Workload Management System

- DIRAC API interfaces with the client securely
  - Uses DISET Framework
- Data Optimizer queries the LCG File Catalogue (LFC) to determine suitable sites for job execution
- Agent Director and Agent Monitor services handle submission to LCG
  - Pilot Agents go through the LCG Resource Broker as normal jobs

DIRAC Agent on WN
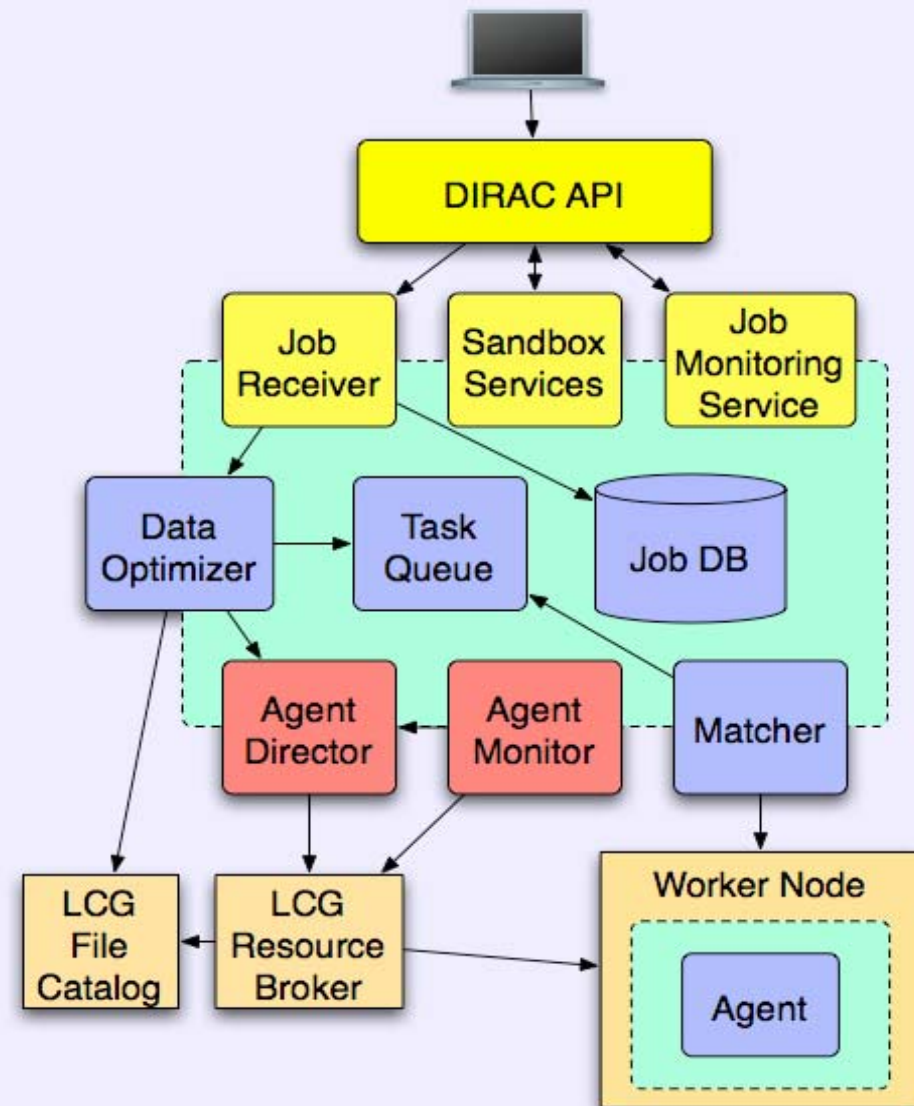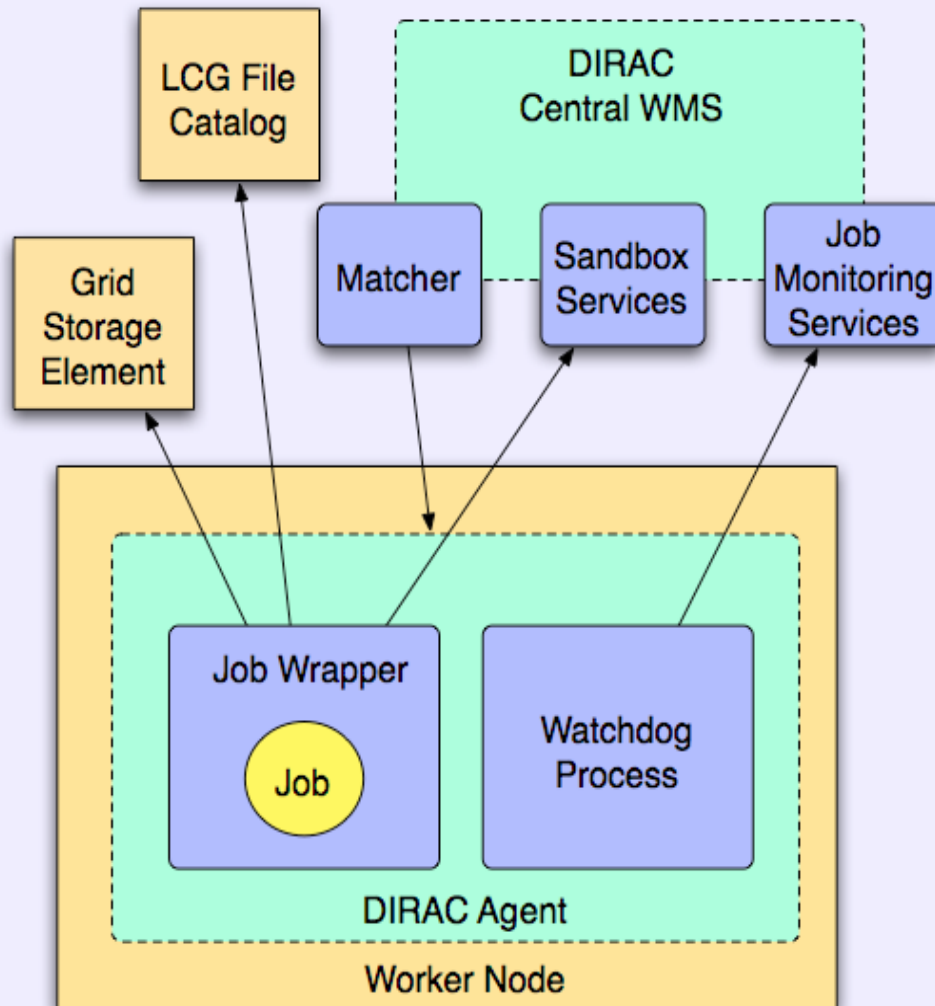
- Dynamically creates Job Wrapper
  - Handles Input/Output Sandboxes
  - Transfers and registers Output Data as required, according to LHCb conventions
- Sends accounting and site specific information back to DIRAC WMS
- Runs a Watchdog process in parallel to the application
  - Provides 'heartbeat' for DIRAC Monitoring
- Provides access to any requested Input Data

Comparison of DIRAC Modes

- A study was performed to compare the possible WMS strategies for analysis jobs
  - We know DIRAC can deal with long production jobs, what about the other extreme?
- Consider shortest possible 'useful' job
  - Analysis of 500 Events
- These short jobs are of high priority since results must be back as soon as possible

DIRAC WMS Pilot Agent Strategies

- There are several ways to use the DIRAC infrastructure but the end goal is to minimise the start time of user analysis jobs
  - The following explores some of the possibilities
- DIRAC Modes of submission
  - 'Resubmission'
    - Pilot Agent submission to LCG with monitoring
    - Multiple Pilot Agents may be sent in case of LCG failures
  - 'Filling Mode'
    - Pilot Agents may request several jobs from the same user, one after the other
  - 'Multi-Threaded'
    - Same as 'Filling' Mode above except two jobs can be run in parallel on the Worker Node

Experiments Performed

- Short (5 Mins) analysis jobs running at Tier 1 Centres
- Jobs for different modes were intermixed in order to ensure the same Grid 'weather'
- Submission of jobs was at the pace of the Resource Broker
- Job start time is defined as the time between submission and the application starting on the WN
- For each experiment 3 Users submitted 100 jobs
  - Each user submitted jobs for one of the three different modes described above
- These experiments were repeated several times to see if results are reproducible

Start Times for 10 Experiments, 30 Users

**Start Times for 30 Users
3000 Jobs, 1.5 Million Events**

Legend:
- Resubmitting
- Filling Mode
- Multi-Threaded

X-axis: **Start Time (Mins)**

Start Times for 10 Experiments, 30 Users

Start Times for 30 Users
3000 Jobs, 1.5 Million Events

LCG limit on start time, minimum was about 9 Mins

Legend:
- Resubmitting
- Filling Mode
- Multi-Threaded
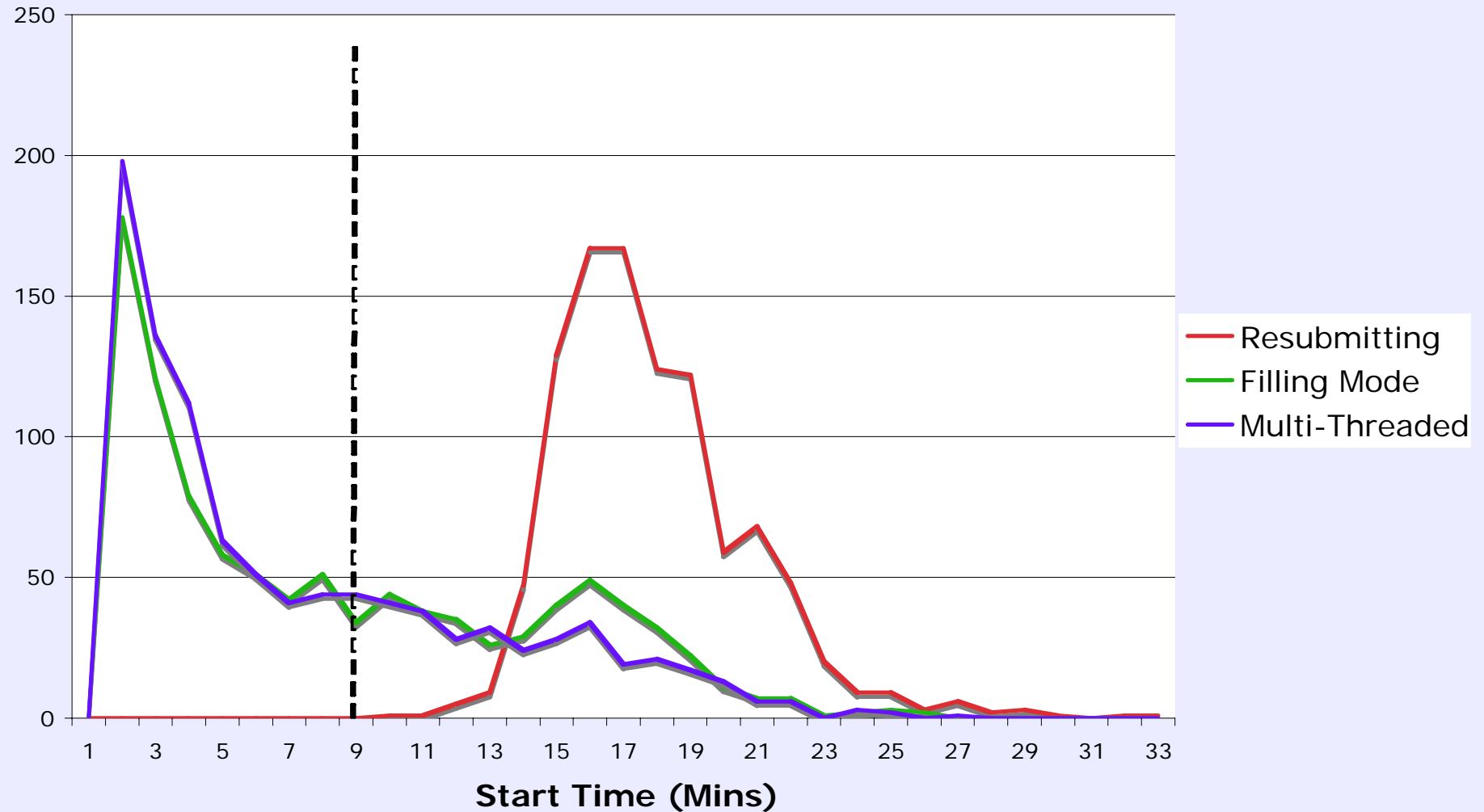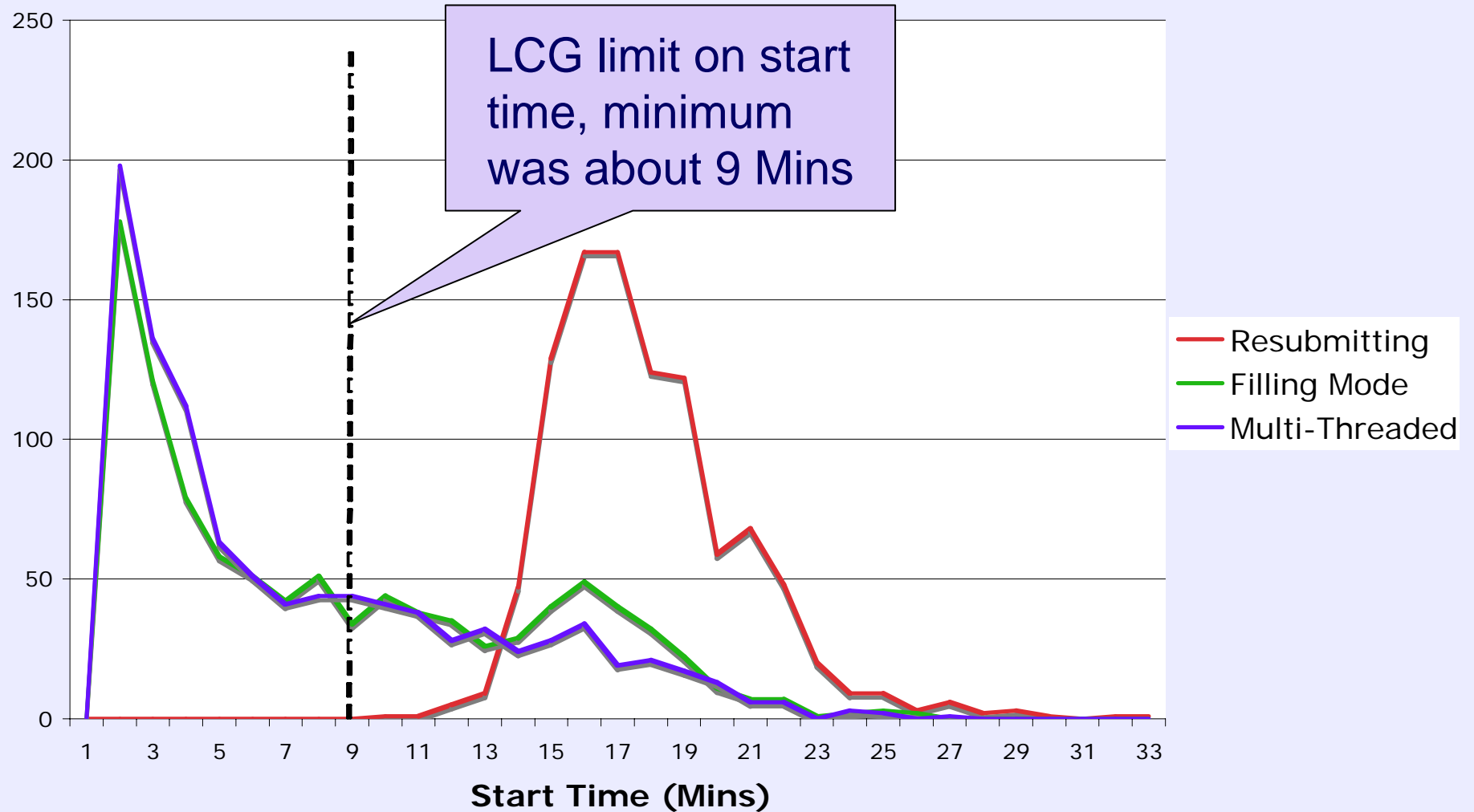
X-axis: Start Time (Mins)

Mean Start Times for 10 Experiments with 30 Users
3000 Jobs, 1.5 Million Events

Analysis of Results
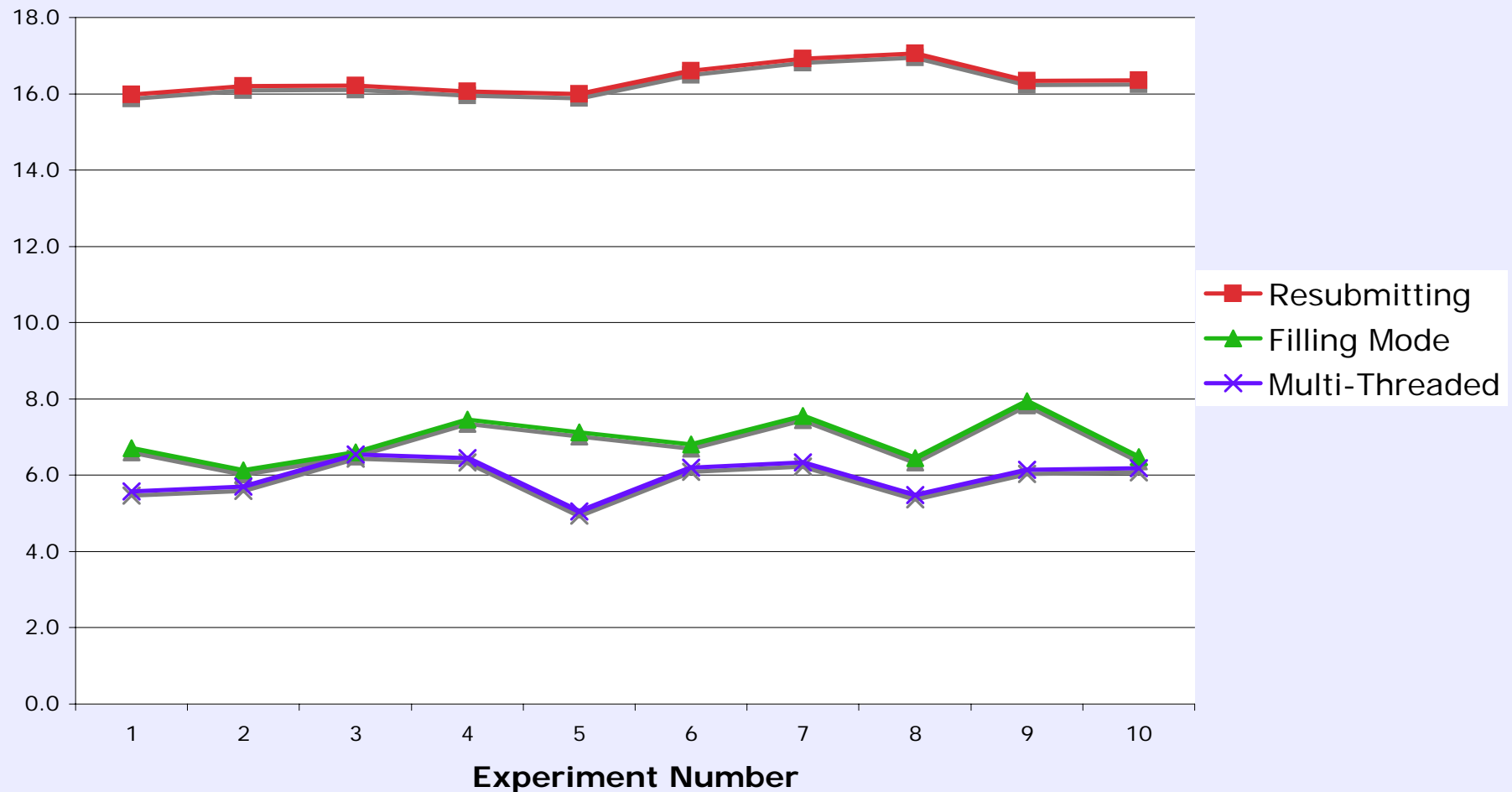
- All 3000 jobs successfully completed within ~40 Mins of starting
- Can see clear improvement on start times for Multi-Threaded and Filling modes
  - Even when LCG performance is optimal, see significant reduction
  - A factor of 3 to 4 fewer LCG jobs need to be sent
    - These modes therefore reduce the load on LCG

- Optimization of workload is performed for each user independently
  - Here each user submitted 100 jobs
    - This can be further improved by optimizing the workload on the level of the Virtual Organisation (VO)

Added Value of DIRAC for Users

- DIRAC masks the inefficiencies of LCG
  - In these tests LCG performed well, this is not always the case…
  - Can execute LHCb VO policy in one central place for the benefit of all users
    - e.g. troublesome LCG sites can be banned for all users at once
- DIRAC WMS currently demonstrated ~2Hz throughput for analysis jobs
  - Actual throughput limited by available LCG resources and capacity of Resource Broker

- The DIRAC API provides a simple yet powerful tool for users
  - Access to LCG resources is provided in a simple and transparent way
- DIRAC Multi-Threaded and Filling modes show significant reductions on the job start times
  - Also reduce the load on LCG
- Workload management on the level of the user is effective
  - Can be more powerful on the level of the VO
- DIRAC infrastructure for distributed analysis is in place
  - Now have real users

Further Information

- DIRAC Infrastructure for Distributed Analysis, Paterson, S., Tsaregorodtsev, A., CHEP06, Mumbai, India, 13-17 February, 2006.

- DIRAC, the LHCb Data Production and Distributed Analysis system, Andrei Tsaregorodtsev et al., CHEP06, Mumbai, India, 13-17 February, 2006.

- Experience with Distributed Analysis in LHCb, Ulrik Egede et al., CHEP06, Mumbai, India, 13-17 February, 2006.

- DIRAC Security Infrastructure, Casajus Ramo, A., Graciani Diaz, R., CHEP06, Mumbai, India, 13-17 February, 2006.

- DIRAC Distributed Analysis on the Computing Grid, Paterson, S., Tsaregorodtsev, A., LHCb Note 2005-072, December, 2005.

Backup Slide - Workload Optimization

Comparison of 1 and 10 Users for Multi-Threaded Mode
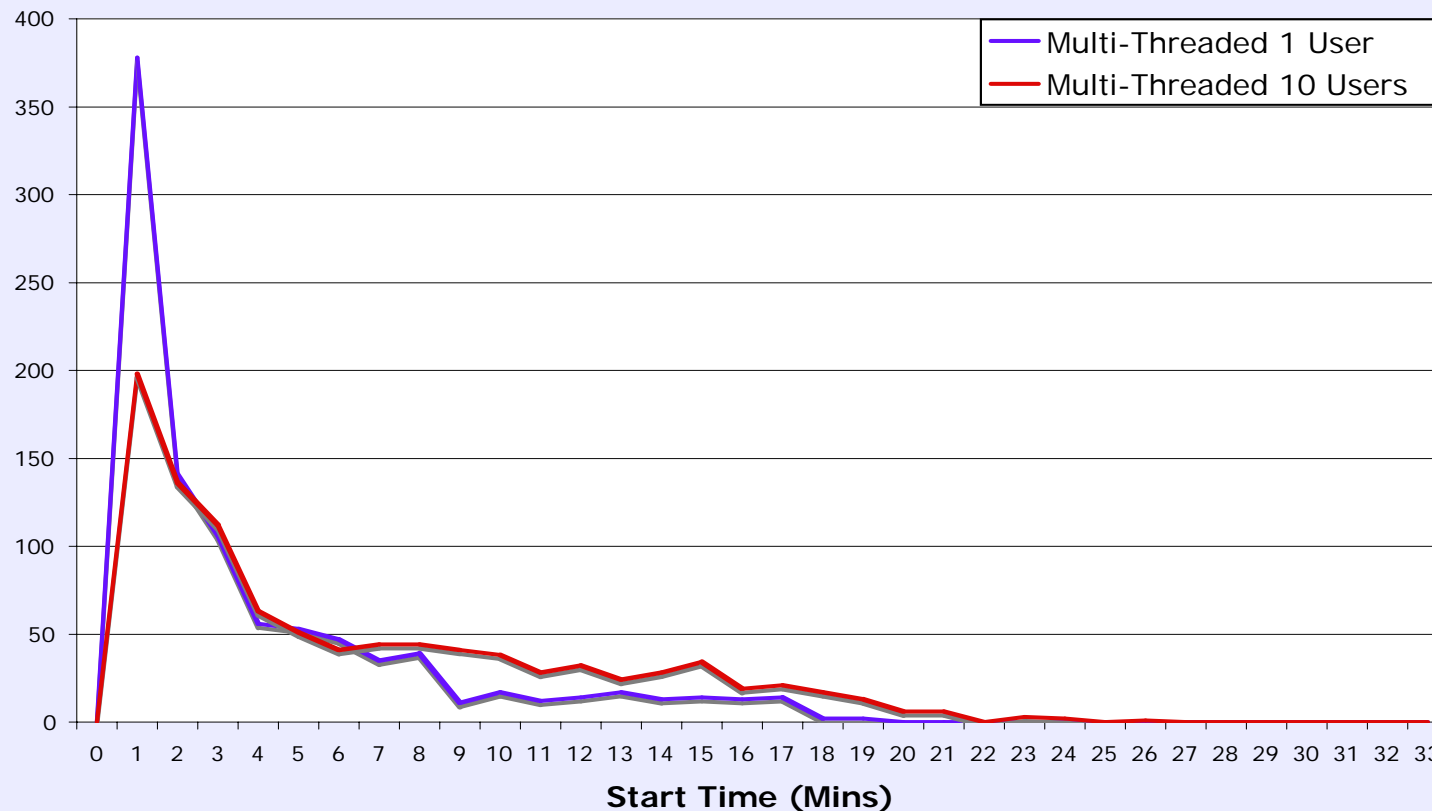
- Two cases,
  - 1 user submitting 1000 jobs
  - 10 users submitting 100 jobs

Backup Slide - Analysis of Results

- There are clear gains in efficiency when the number of users is reduced

- Thus, workload optimization per user is effective but less effective than optimization per Virtual Organisation could be

Total times for 30 users
3000 Jobs, 1.5 Million Events