

# *Monitoring in JIM*

## *Monitoring needed for D0 data reprocessing:*

Jobs were monitored manually in the previous reprocessing, this was very time consuming. With the increased amount of data we must have an automated solution.

## *Existing Monitoring solutions:*

At Nikhef:

- Information from Jeff Templon (wuppertal presentation + email)
- R-GMA based solution (SQL interface, proposed grid standard, employs MySQL)

At Lyon:

- Waiting for info from Patrice and Tibor

## *What's missing*

- A solution that can be used in every JIM farm without additional installations. This can be made to work since JIM includes a DB with network tools, namely XMLDB.
- May need to expand the succesful existing designs for usage on multiple farms.



# *From Templon's Talk*

## **R-GMA at 0300:**

- the producers are the jobs – producing info “I landed on a worker node at NIKHEF at 10.38.22”
- Consumer is the “archiver” – snarfs all D0 records and puts them into a back-end (oracle, mysql) DB for posterity
- Site servers allow communication between WN pools (private IP space) and outside world ... producers and consumers actually live on these machines
- Registry is marketplace of who has what info and who is looking for what – bring producer & consumer into direct contact.

## **R-GMA at 1400:**

- Consumer is physicist: tell me all jobs ending in failure during last 24 hours (so I can figure out why, and then resubmit)
- Producer is the “archiver” – serves records from back-end DB onto public R-GMA net upon request. Also can do table joins, very handy!



# Templon's DB (I)

Table=d0jsu4

```
Schema=[CREATE TABLE d0jsu4 (  
  jobID VARCHAR(64) PRIMARY KEY,  
  submit_time INT,  
  ui_node VARCHAR(64),  
  submitter VARCHAR(128))]
```

**Job submission** (who, when, where)

Table=d0jst4

```
Schema=[CREATE TABLE d0jst4 (  
  jihash VARCHAR(22) PRIMARY KEY,  
  jobID VARCHAR(64),  
  start_time INT,  
  site VARCHAR(64),  
  command VARCHAR(255))]
```

**Job execution starts** (when, where, how)

The "jihash" field is created from a hash of the grid job ID plus worker node plus timestamp, to create a unique identifier. It's needed because sometime a failed job gets run again on a different system, so the grid job identifier is not necessarily a unique key for any table except "submit".

Table=d0pst4

```
Schema=[CREATE TABLE d0pst4 (  
  jihash VARCHAR(22) PRIMARY KEY,  
  start_time INT,  
  worker_node VARCHAR(64),  
  inp_lfn VARCHAR(255),  
  process_id INT,  
  cpu_type VARCHAR(64),  
  cpu_freq REAL,  
  phys_mem INT,  
  swap_space INT)]
```

Environment setup is done,  
and **processing starts**



# Templon's DB (II)

Table=d0jen4

```
Schema=[CREATE TABLE d0jen4 (  
  jihash VARCHAR(22) PRIMARY KEY,  
  end_time INT,  
  cpu_time INT,  
  wall_time INT,  
  success_code VARCHAR(32),  
  procevt INT,  
  out_lfn VARCHAR(255),  
  phys_file VARCHAR(255),  
  guido VARCHAR(42))]
```

**Job Ends**

When the executable terminates and the file is saved to grid storage, the "job end" record is published. Finally all critical commands are wrapped in a shell that just passes thru all output if everything is OK, but if it isn't it publishes some relevant information to a "error" table.

Table=d0cmd4

```
Schema=[CREATE TABLE d0cmd4 (  
  jicmdhash VARCHAR(22) PRIMARY KEY,  
  jihash VARCHAR(22),  
  cmd_start_time INT,  
  cmd VARCHAR(255),  
  exit_stat INT,  
  output VARCHAR(255))]
```

**Error wrapper reporting code**

*Questions:*

- lfn = long file name? What are guido & phys\_file?
- Why do we need 3 tables with the same primary key?

