



DTM a lightweight computing virtualization system based on iRODS

Yonny Cardenas,
Pascal Calvat, Jean-Yves Nief,
Thomas Kachelhoffer

▶ Overview



- Introduction
- User Operation
- Architecture
- An iRODS Based System
- Implementation and Evaluation
- Computing workload managed by data activity
- Future work

▶ Introduction



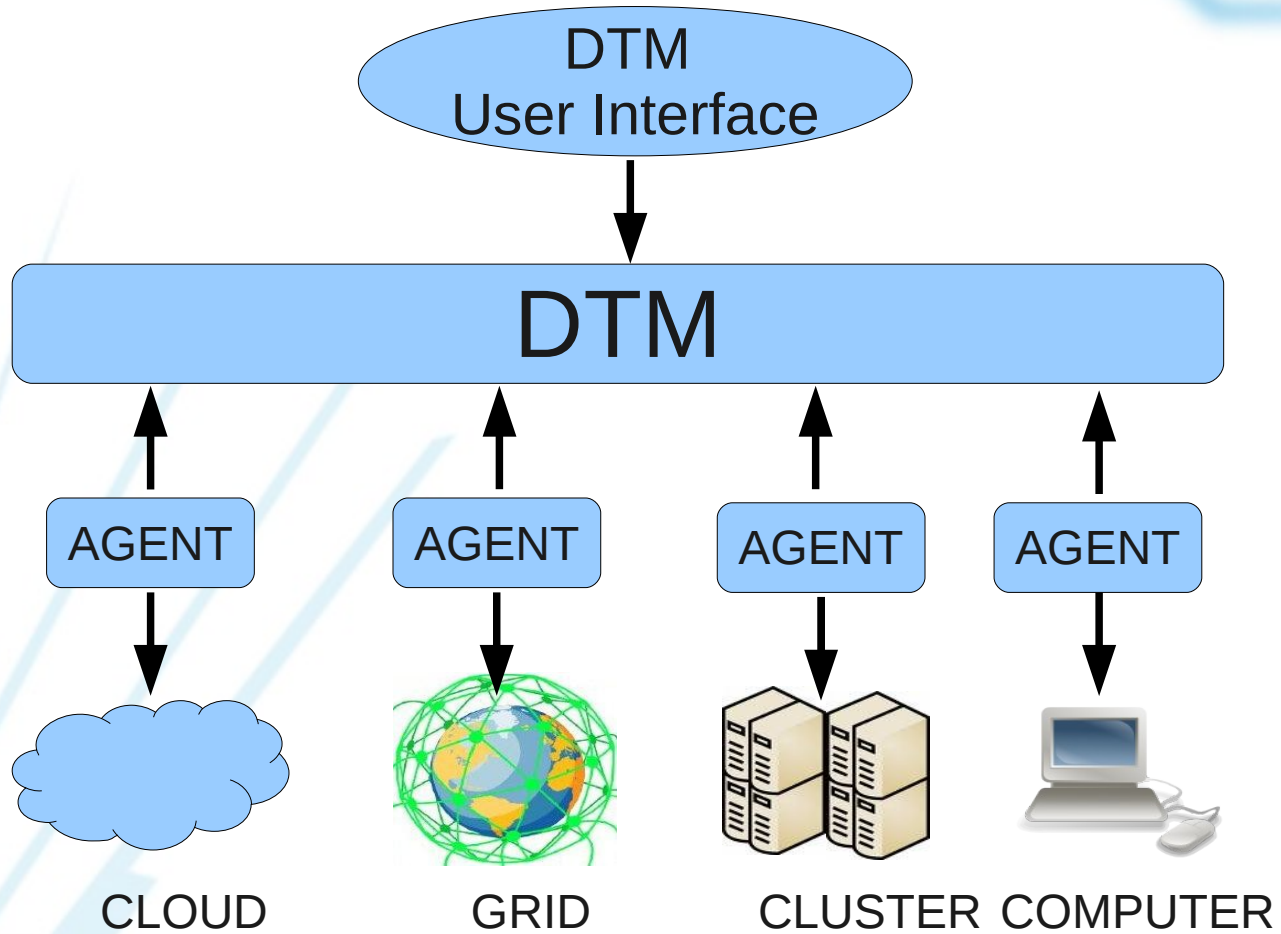
- Users need to employ several computing infrastructures simultaneously: batch clusters, grid computing, clouds, ...
- They want automate the utilisation procedures to improve efficiency of use of the resources
- But these infrastructures use different middleware and interfaces

▶ Introduction



- We propose the Distributed Task Manager (DTM)
- With DTM, users have the perception of using a single, simplified and powerful computing system
- DTM creates simple and uniform resource access across multiple heterogeneous computing platforms and provides location transparency
- It uses a pull scheduling approach to execute tasks in a distributed way via agents that hide the infrastructure heterogeneity

Introduction

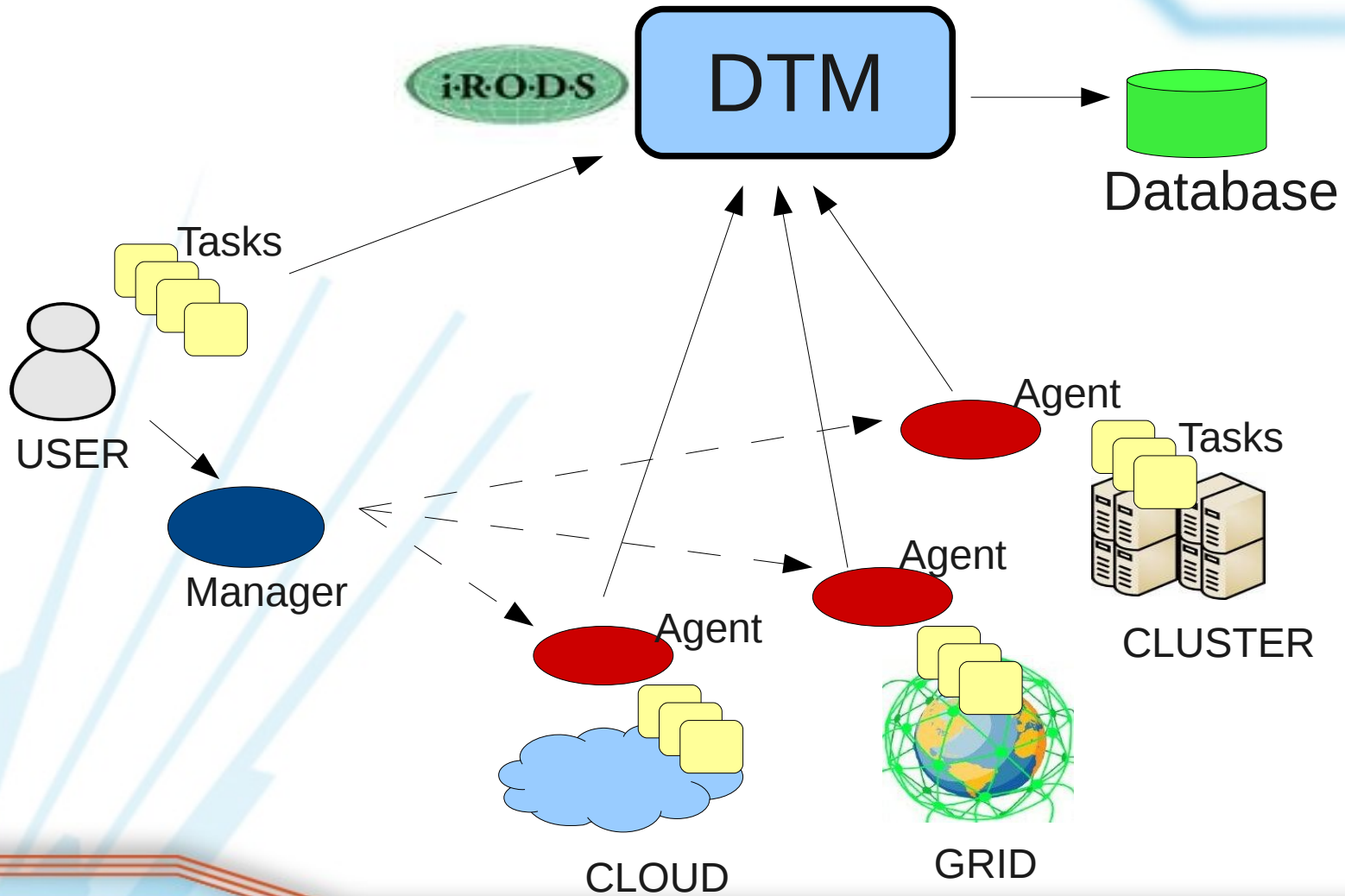


▶ User Operation



- Register tasks and productions
 - Specify the basic task requirements (CPU, memory, ...)
 - A task must belong to a production
- Production enabling
 - Mark tasks as ready to consume
- Production performing
 - Perform a production on subset of available Infrastructures. Launches manager and agents

User Operation



▶ User Operation



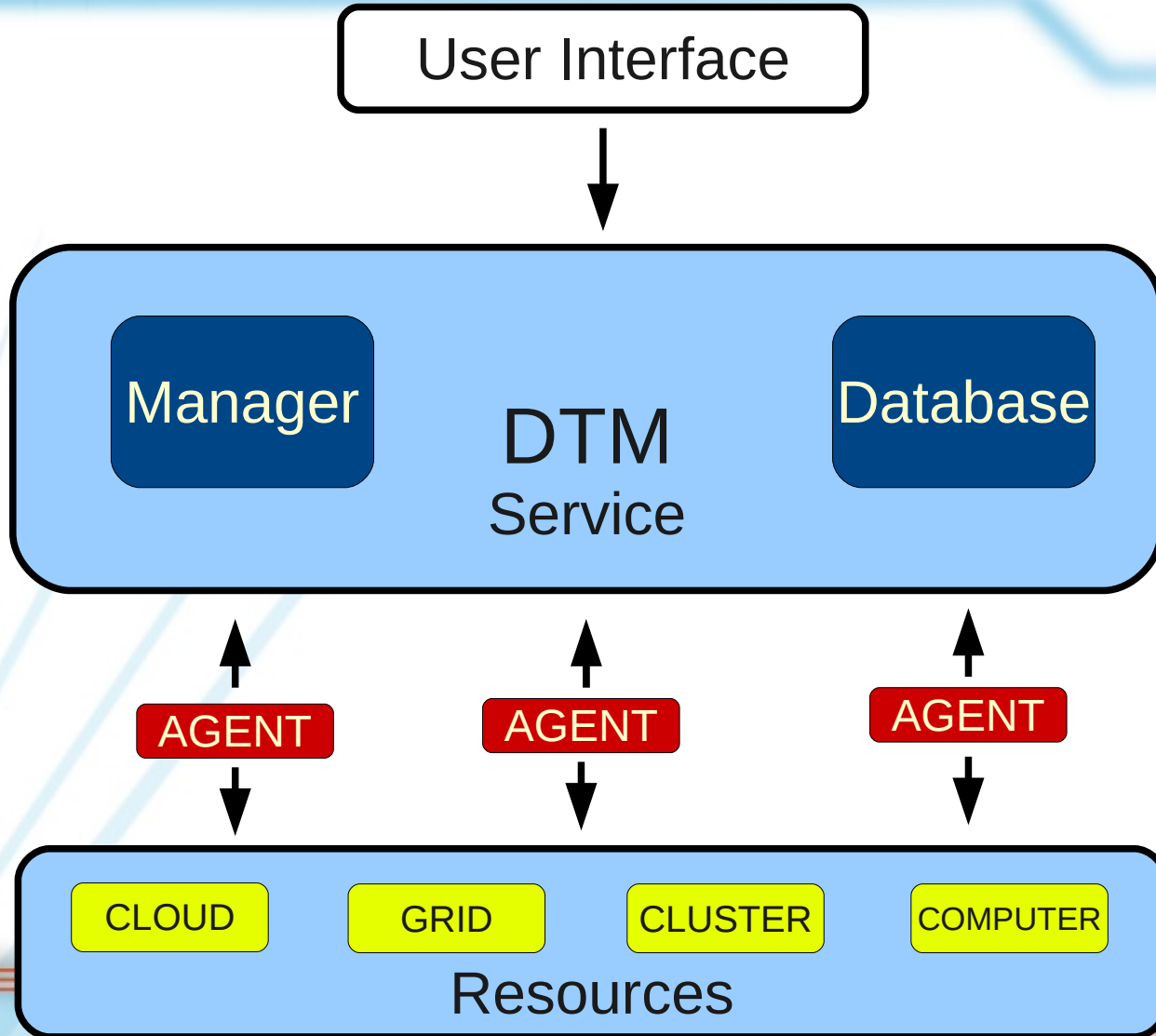
```
cardenas@localhost:~  
File Edit View Terminal Help  
$ dtm-task-add -h  
usage: dtm-task-add -c cputime -m memory -p production -s script -t taskname  
       [-a scriptarg] [-d dependency] [-b tmpbatch] [-v version][-h help]  
  
REQUIRED OPTIONS:  
-c,--cputime <cputime>      Max cpu time for this task in HS06 seconds  
-m,--memory <memory>       Max memory for this task in Megabytes  
-p,--production <production> Production name  
-s,--script <script>       Script file to be executed  
-t,--taskname <task name>  Task name must be unique for a given  
                             production  
  
OPTIONAL OPTIONS:  
-a,--scriptarg <scriptarg>  Quoted args for the script be executed  
-d,--dependency <dependency> Production name dependency  
-b,--tmpbatch <tmpbatch>    Max tmpbatch size for this task (BQS)  
-h,--help                    print this message  
-v,--version                  print the version and copyright  
  
dtm : Distributed Task Manager 1.1  
$
```


Architecture



- **Database:** Information system
 - Registers state information and actions on tasks.
 - It ensures the integrity of information related with task assignment
 - Implements transactions and operations as stored procedures
- **Task:** Action that a user wishes to perform on a computing machine
 - An user creates a production to group a set of relatively homogeneous tasks
- **Agent:** Executes one or several user tasks
 - It runs as independent job or process
- **Manager:** Coordinates the execution of tasks and agents.

Architecture

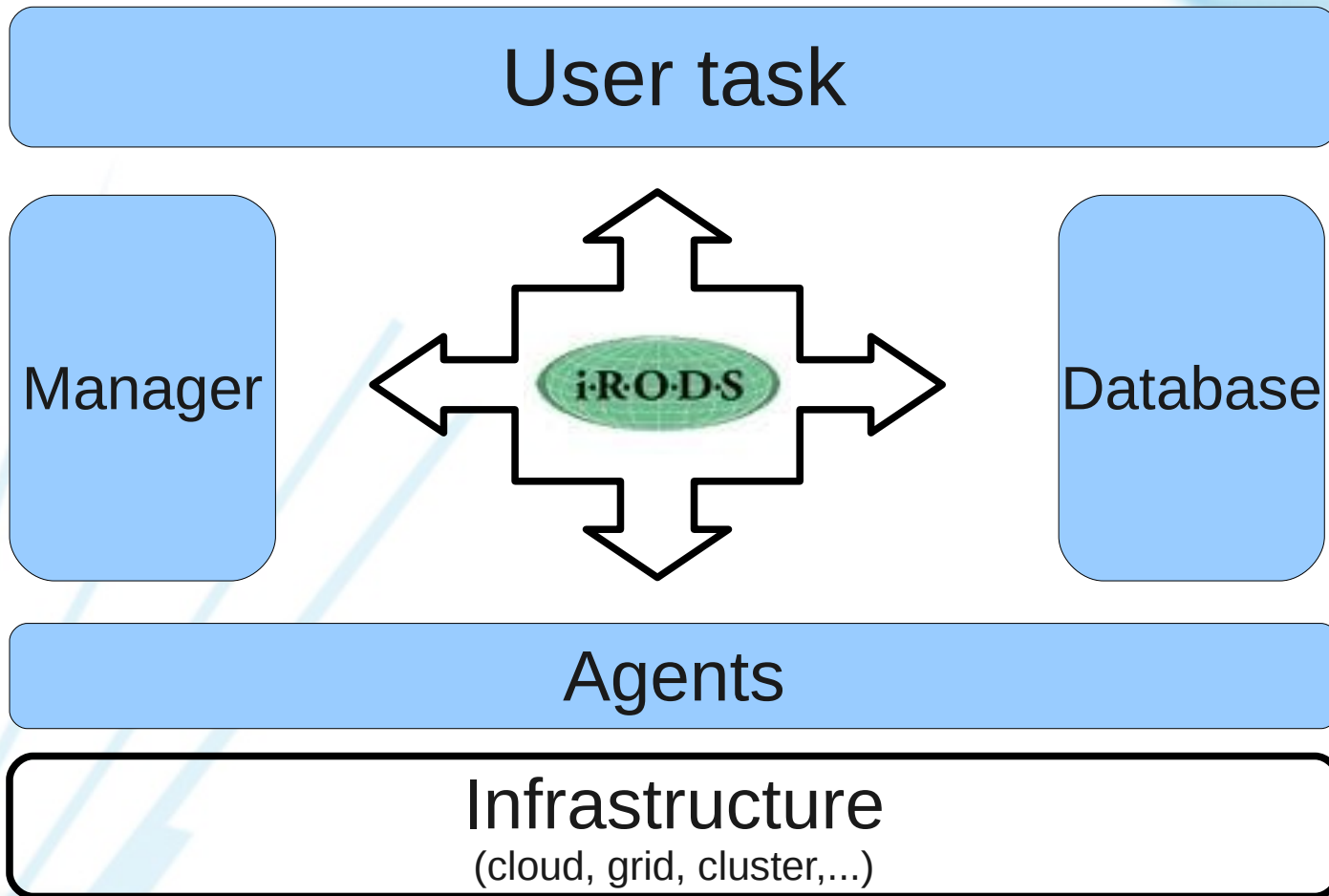


An iRODS Based System



- **Authentication**
 - User authentication and across systems are hidden
 - Password and GSI (EGI)
- **Rule-oriented Database Access (RDA)**
 - Transparent access to the DTM database on server
 - MySQL stored procedures invoked from RDA micro-services
- **Remote rule execution**
 - Core business logic is implemented as rules on the server side
 - Clients and components initiate DTM actions that invoke rules on iRODS servers.

An iRODS Based System



Implementation



- Support:
 - Grid: gLite (EGI)
 - batch: GE, BQS, Torque (coming)
- **Database:** mySQL stored procedures
 - 214 MySQL code lines
- **16 bash shell scripts**
 - Manager: 303 lines (job)
 - Agent: 383 lines (job)
 - Unix/Linux user command-line utilities (eleven commands)
 - All: 2642 lines
- **15 iRODS rules**
- The full implementation is **124 Kbytes** of total size

Implementation



```
cardenas@localhost:~/work/working/dtm
File Edit View Terminal Help
$ wc -l * | sort -n
 59 dtm-task-monitor.sh
 65 dtm-prod-list
 74 README.TXT
 76 dtm-proxy-init
 76 dtm-task-list
 87 dtm-init
 87 dtm-prod-enable
102 dtm-start
121 dtm-task-set-status
146 dtm-task-get-next
170 dtm-shell-functions.sh
175 dtm-task-add
189 dtm-manager-input-validation.sh
216 dtm-cancel
303 dtm-manager
313 dtm-cron-functions.sh
383 dtm-agent.sh
2642 total
$ du -hs
124K  .
$
```

Implementation



```
cardenas@localhost:~/work/working/paper/presentatio
File Edit View Terminal Help
mysql> SELECT ROUTINE_TYPE, ROUTINE_NAME FROM INFORMATION_SCHE
+-----+-----+
| ROUTINE_TYPE | ROUTINE_NAME |
+-----+-----+
| PROCEDURE    | agentcountnotreported |
| PROCEDURE    | agentcountrunning |
| PROCEDURE    | logadd |
| PROCEDURE    | nexttask |
| PROCEDURE    | nexttaskclone |
| PROCEDURE    | proddelete |
| PROCEDURE    | prodenable |
| PROCEDURE    | prodlist |
| PROCEDURE    | receivercheck |
| PROCEDURE    | receiverconfig |
| PROCEDURE    | taskadd |
| PROCEDURE    | taskcount |
| PROCEDURE    | taskcountrunning |
| PROCEDURE    | taskcountwithoutreport |
| PROCEDURE    | taskdelete |
| PROCEDURE    | taskfailedrestart |
| PROCEDURE    | taskhistorytransfer |
| PROCEDURE    | tasklist |
| PROCEDURE    | tasknotrespondingrestart |
| PROCEDURE    | taskpriority |
| PROCEDURE    | tasksetstatus |
| PROCEDURE    | tasksumcpurequested |
+-----+-----+
22 rows in set (0.00 sec)
```

Implementation



- User Invocation (bash shell):

```
IRULE_NAME="acDtmTaskGetNext(*P,*C,*S,*J,*H,*A,*U,*M,*O)"
```

```
IRULE_IN="*P=$PROD_ID%*C=$CPU_TIME_LEFT%*S=$THE_SITE  
%*J=$JOB_NAME%*H='$HOSTNAME'%*A=$HOSTADDRESS  
%*U='$USER'%*M='$MAXREPLY'%*O=null"
```

```
IRULE_OUT="*O"
```

```
irule -v "$IRULE_NAME" "$IRULE_IN" "$IRULE_OUT"
```

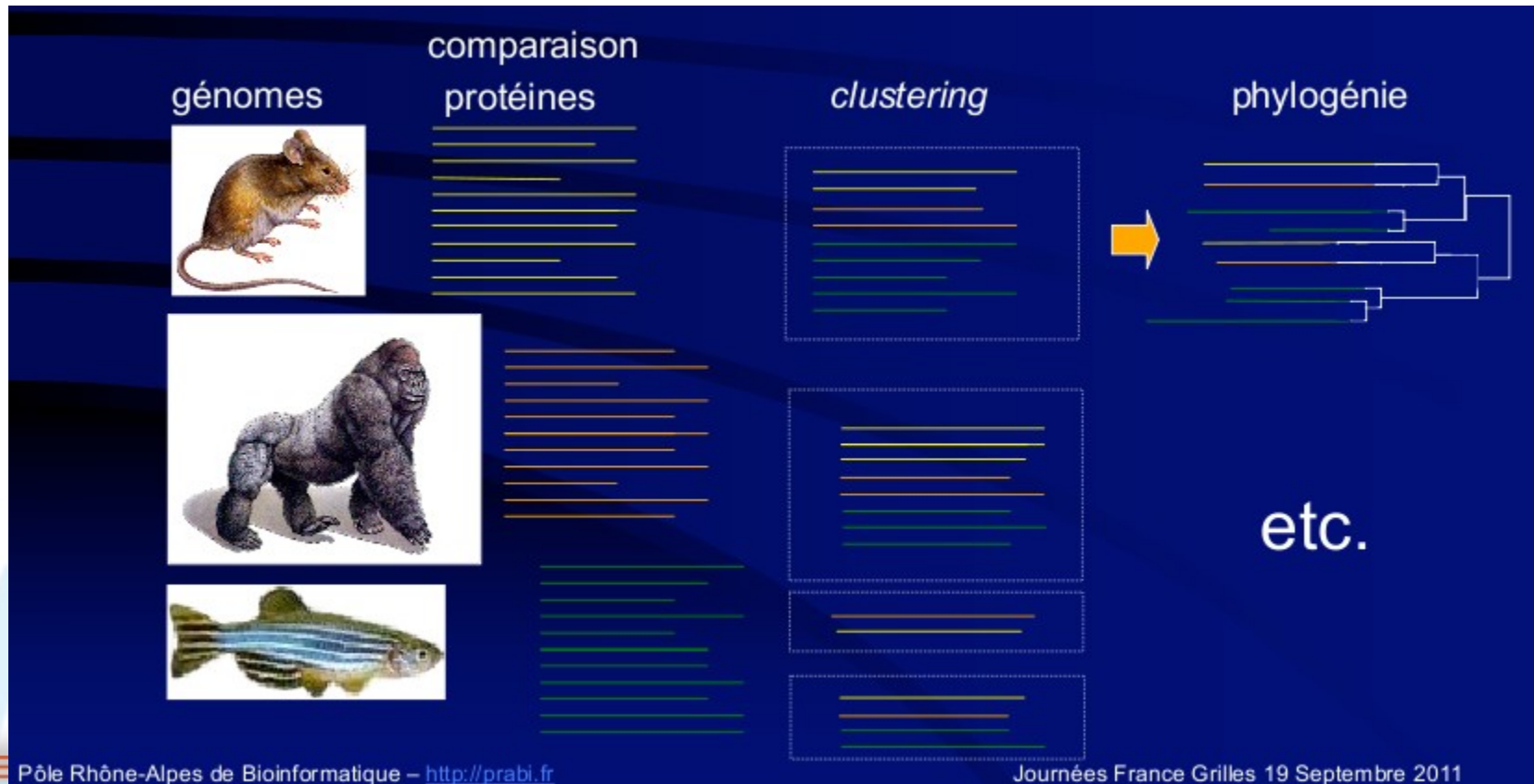
- Rule at iRODS server side:

```
acDtmTaskGetNext(*P,*C,*S,*J,*H,*A,*U,*M,*O)||assign(*R,  
userNameClient)##msiRdaToString(RDA,"call  
nexttask(*R',*P,*C,*S,*J,*H,*A,*U,*M)",null,null,null,null,*O)|nop
```


Evaluation



- Computational genomics (phylogenetic analyses) S. Penel
- Simultaneous execution of thousands of job agents



Portal RunMyCode



Portal Run My Code - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.runmycode.org

Portal Run My Code

Register | Sign In

Search here ... Search

Home
First visit?
Submit your code

Search by themes
Advanced search

Help/FAQ
Our partners
Contact us

A revolutionary scientific tool

Research diffusion

RunMyCode allows researchers to spread their research **globally**.

Latest scientific methods

RunMyCode allows people to implement the latest scientific methods in a **user-friendly environment**.

Validation tool

RunMyCode allows researchers to replicate scientific results in order to check their **validity and robustness**.

About Concept **Purpose**

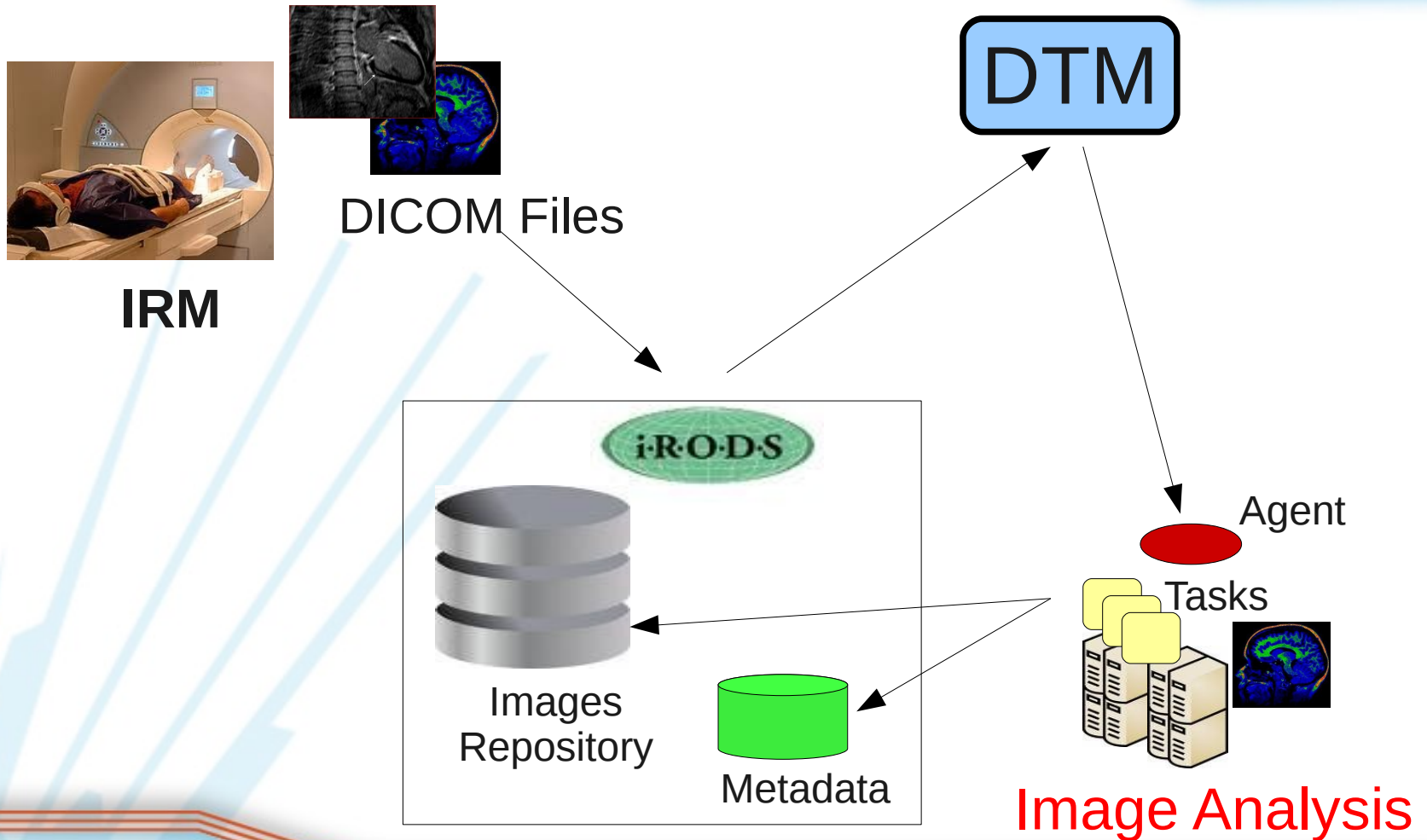
Create your own companion website >>

▶ Data activity: Use Case



- Computing workload managed by data activity
- Integration between DTM and iRODS
- Triggering of a batch or grid job submission after data update
- Automatic processing of DICOM files
 - Standard format for information in medical imaging
- Analysis: Indexing of content and metadata from image files.

▶ Data activity: Use Case



▶ Future work



- Replace the RDA interface with the iRODS database resource
- Update to new rule syntax
- Grid job submission: Replace the gLite user interface with JJS/JSAGA
- Support automatic submission on clouds handled by the DTM manager
- Job submission based on data management policies:
 - task execution location that integrates criteria related to data placement



Thank you

Questions ?