DTM a lightweight computing virtualization system based on iRODS

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

IRODS User Group Meeting, Tucson, AZ, USA, Mars 1-2, 2012
Overview

- Introduction
- User Operation
- Architecture
- An iRODS Based System
- Implementation and Evaluation
- Computing workload managed by data activity
- Future work
• Users need to employ several computing infrastructures simultaneously: batch clusters, grid computing, clouds, ...

• They want to 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

DTM User Interface

DTM

AGENT

AGENT

AGENT

AGENT

CLOUD

GRID

CLUSTER COMPUTER
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
  Tasks
  Manager
  Agent
  Agent
  Agent
  CLUSTER
  CLOUD
  GRID
  Database

DTM

i·RO·DS
User Operation

```bash
$ 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.
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

User task

Manager  iRODS  Database

Agents

Infrastructure
(cloud, grid, cluster,...)
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

```
$ 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
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

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
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

- DTM
  - Agent
  - Tasks
  - Image Analysis

- Images
- Metadata

- IRM
- DICOM Files
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?