WMS (Workload Management System) и запуск заданий

Олешко С.Б.
Петербургский институт ядерной физики
г.Гатчина

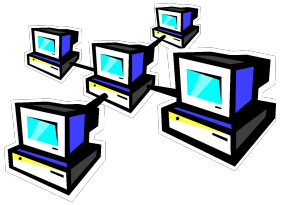
Использование заданий для запуска приложений

- **Задание (job) – это средство для запуска приложений в Грид**
- **Информация, которая должна быть определена, когда задание должно быть запущено в Грид**
  - Характеристики задания
  - Требования задания и условия на вычислительные ресурсы
    - Включая требования на программное обеспечение
  - Требования к данным
- **Эта информация определяется при помощи Job Description Language (JDL)**
  - Основан на *CLASSified ADvertisement language (ClassAd) из проекта Condor*
    - *ClassAd – последовательность атрибутов, разделённых (;)*

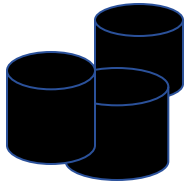
Как это работает? Основные компоненты.

User Interface (UI): Место, откуда пользователь вошёл в Грид

Resource Broker (RB): Сопоставляет пользовательские требования и доступные ресурсы Грид



Computing Element (CE): Очередь на выполнение заданий на том кластере, где будет выполняться задание



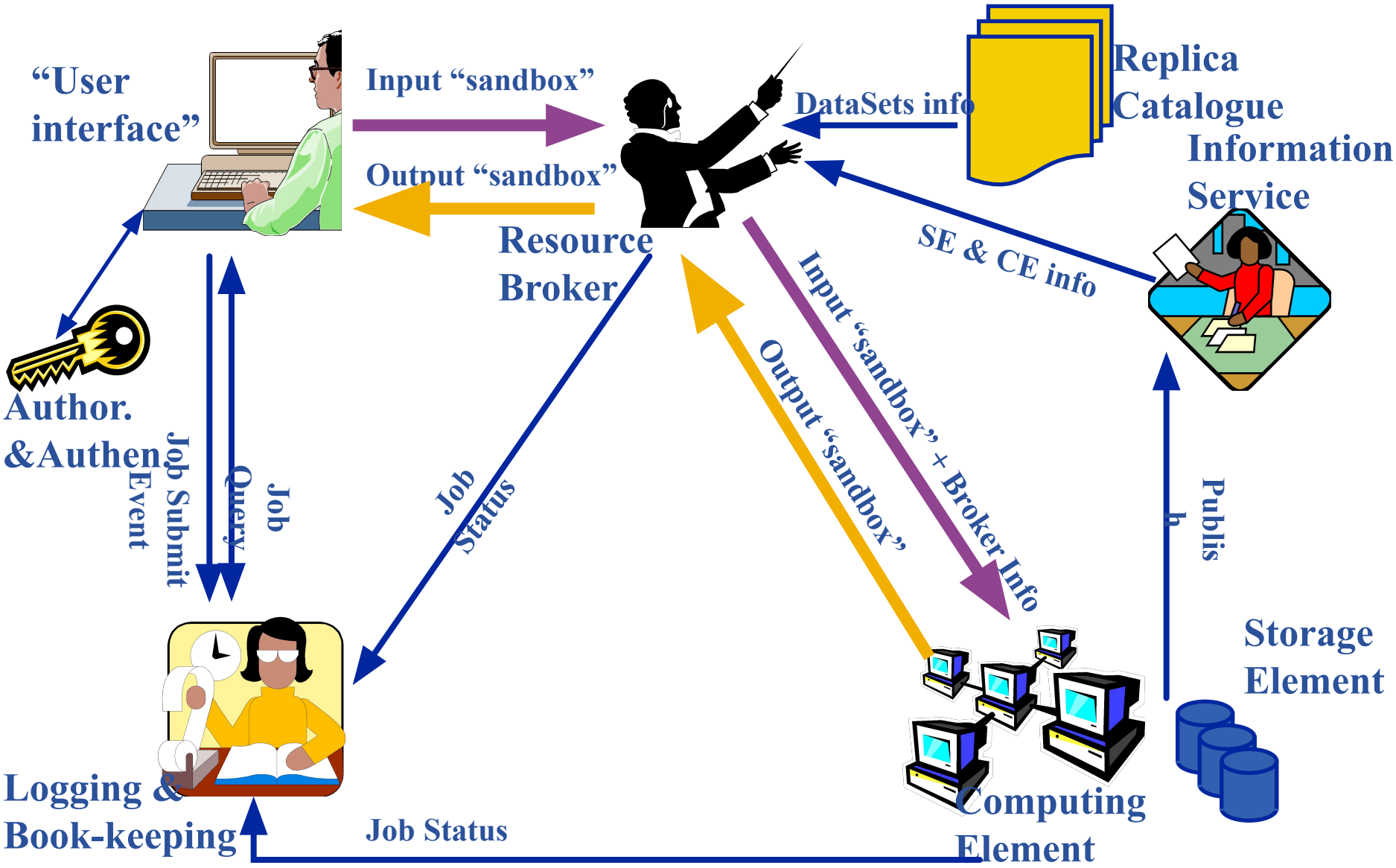
Storage Element (SE): Сервер хранения данных, где сохраняются Грид файлы (чтение/запись/копирование) или их реплики.

Information System: Характеристики и статус для CE иSE (Используя “GLUE schema”)

EGEE/LCG Workload Management System

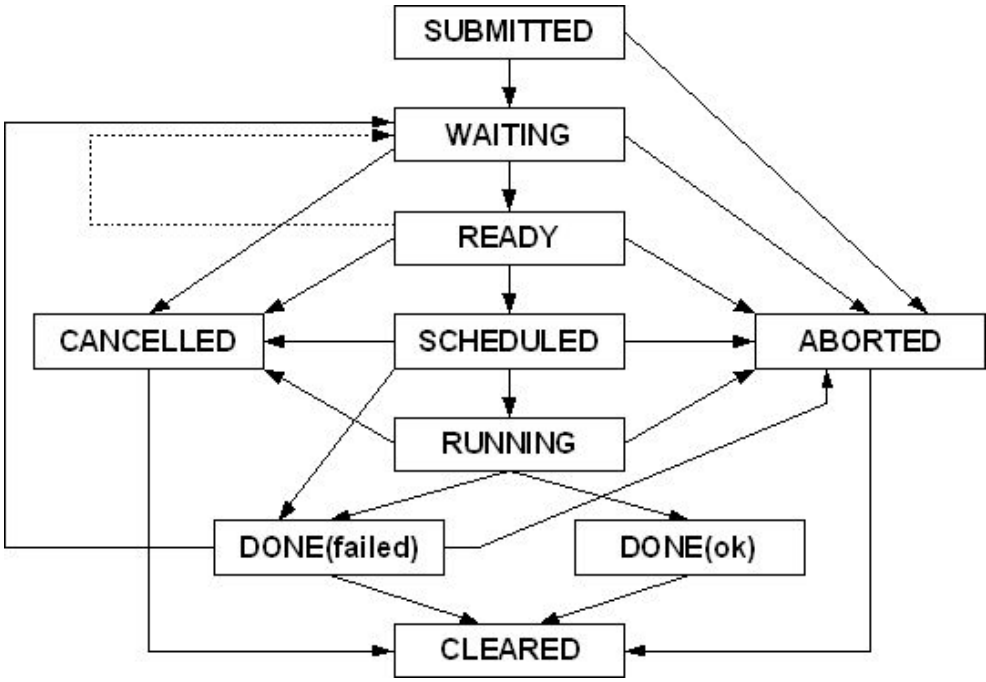
- **Пользователь управляет заданиями через подсистему управления загрузкой (Workload Management System - WMS);**
- **Основная задача WMS - планирование и управление распределенными ресурсами в системе Grid;**
- **Что может пользователь?**
  - Посылать задачи на выполнение;
  - Выполнять задачи на наиболее подходящих для этого ресурсах (WMS автоматически оптимизирует использование ресурсов);
  - Получать информацию о состоянии задач;
  - Получать результаты выполнения задач.

Сервисы и запуск заданий

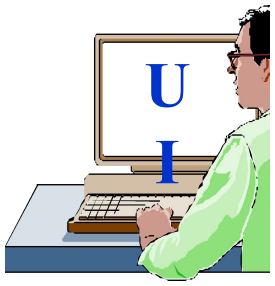


Выполнение задания

SUBMITTED - задание послано пользователем, но пока не обработано Network Server
WAITING - задание принято Network Server, но ещё не обработано Workload Manager
READY - заданию назначен Computing Element, но оно туда ещё не передано
SCHEDULED - задание ожидает в очереди на Computing Element
RUNNING - задание выполняется
DONE - задание завершилось
ABORTED - задание снято WMS (т.к. слишком долгое, срок действия сертификата истёк, и т.п.)
CANCELLED - задание снято пользователем
CLEARED - Output Sandbox передан на User Interface



Запуск задания

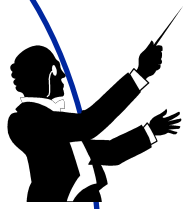


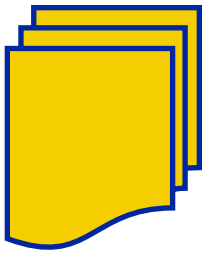
RB узел

Network Server

Workload Manager

Job Contr. - CondorC



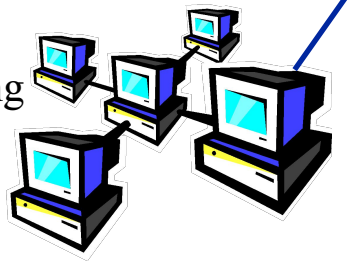


RLS

Information Service

Характеристики. и статус CE

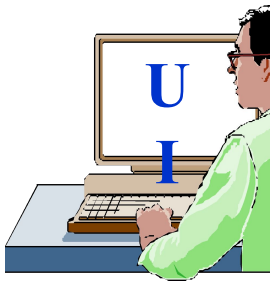
Характеристики и статус SE



Computing Element

Storage Element

Запуск задания



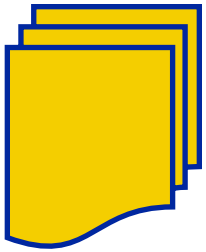
RB узел

Network Server

Workload Manager

Job Contr. - CondorC

UI: позволяет пользователям получить доступ к функциональности WMS (CLI, GUI, C++ и Java APIs)



RLS

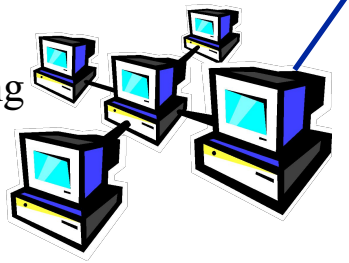
Job Status

submitted

Information Service

Характеристики. и статус CE

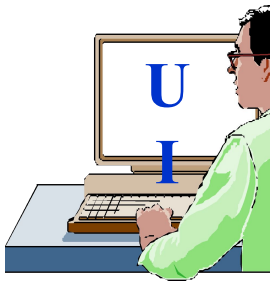
Характеристики и статус SE



Computing Element

Storage Element

Запуск задания



RB узел

Network Server

Workload

RLS

Job Status

submitted

Information

Job Description Language (JDL) определяет характеристики и требования задания

edg-job-submit myjob.jdl

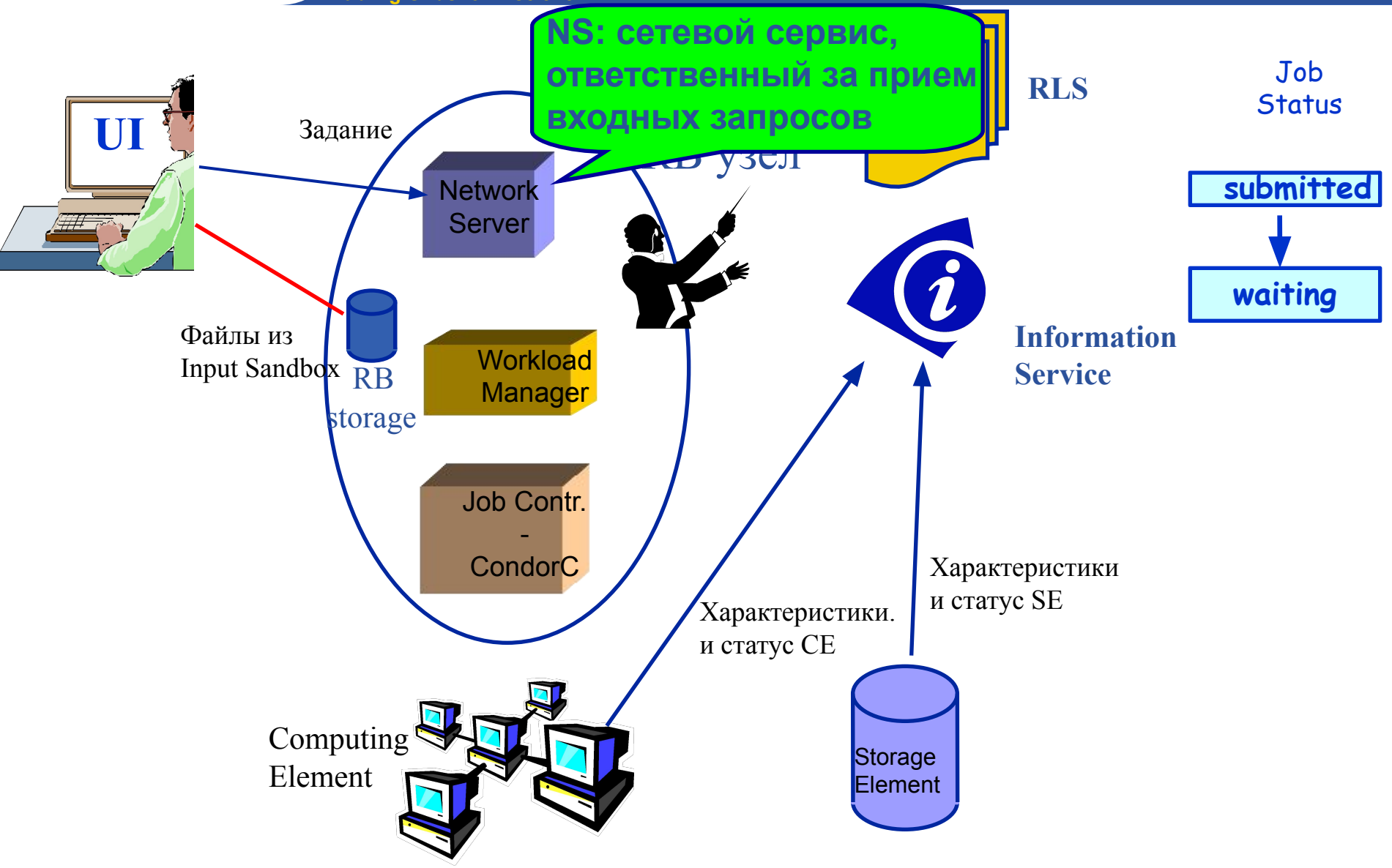
myjob.jdl

```
JobType = "Normal";
Executable     = "$(CMS)/exe/sum.exe";
InputSandbox = {"/home/user/WP1testC","/home/file*", "/home/user/DATA/*"};
OutputSandbox = {"sim.err", "test.out", "sim.log"};
Requirements   = other. GlueHostOperatingSystemName == "linux" &&
other.GlueCEPolicyMaxWallClockTime > 10000;
Rank  = other.GlueCEStateFreeCPUs;
```

Computing Element

Storage Element

Запуск задания



Запуск задания

RB узел

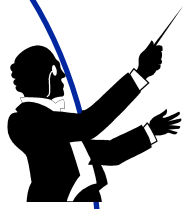
Network Server

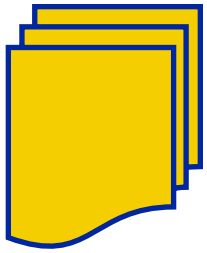
RB storage

Workload Manager

Job Contr. - CondorC

WM: ответственен за выполнение действий для удовлетворения запроса





RLS

Job Status

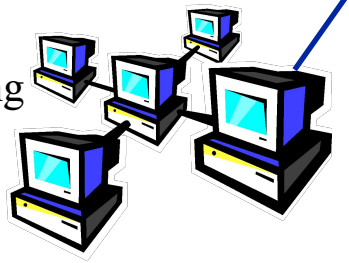
submitted

waiting

Information Service

Характеристики. и статус CE

Характеристики и статус SE



Computing Element

Storage Element

Запуск задания

RB узел

Network Server

Match-Maker/ Broker

RB storage

Workload Manager

Job Contr. - CondorC

RLS

Job Status

submitted

waiting

Где должно выполняться это задание ?

Характеристики. и статус CE

Характеристики и статус SE

Computing Element

Storage Element

Запуск задания

RB узел

Network Server

Match-Maker/ Broker

RB storage

Workload Manager

Job Contr. - CondorC

RLS

Job Status

submitted

waiting

Matchmaker: отвечает за поиск “лучшего” CE, где будет выполняться задание

Характеристики и статус SE

Характеристики. и статус CE

Computing Element

Storage Element

Запуск задания

Где (на каком SE), необходимые данные?

RB узел

Network Server

Match-Maker/ Broker

RB storage

Workload Manager

Job Contr. - CondorC

RLS

Job Status

submitted

waiting

Information Service

Какой статус у Грид-ресурсов

Характеристики. и статус CE

Характеристики и статус SE

Computing Element

Storage Element

Запуск задания

RB узел

Network Server

Match-Maker/ Broker

RB storage

Workload Manager

Job Contr. - CondorC

Выбор CE

RLS

Job Status

submitted

waiting

Information Service

Характеристики. и статус CE

Характеристики и статус SE

Computing Element

Storage Element

Запуск задания

RB узел

Network Server

RB storage

Workload Manager

Job Adapter

Job Contr. -

RLS

Job Status

submitted

waiting

Information Service

JA: ответственен за окончательную “доводку” задания перед процедурой запуска (напр. создание wrapper скрипта, и т.п.)

Computing Element

Element

Запуск задания

RB узел

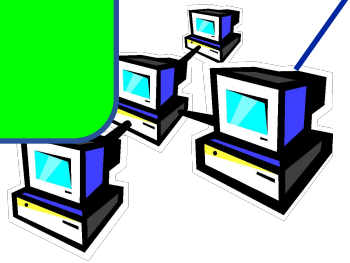
Network Server

RB storage

Workload Manager

Job Contr. - CondorC

JC: ответственен за операции управления заданием (через CondorC)



Element

RLS

Information Service

Job Status

submitted

waiting

ready

Характеристики. и статус CE

Характеристики и статус SE

Storage Element

Запуск задания

RB узел

Network Server

RB storage

Workload Manager

Job Contr. - CondorC

RLS

Information Service

Job Status

submitted

waiting

ready

scheduled

CE получает запрос и посылает задание на выполнение LRMS

Файлы из Input Sandbox

Задание

Характеристики. и статус CE

Характеристики и статус SE

Computing Element

Storage Element

Запуск задания

RB узел

Network Server

RB storage

Workload Manager

Job Contr. - CondorC

RLS

Information Service

LRMS управляет выполнением задания

Job Status

submitted

waiting

ready

scheduled

running

Доступ к данным “через Грид”

Computing Element

Storage Element

Запуск задания

RB узел

Network Server

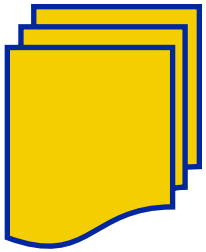
Workload Manager

Job Contr. - CondorC

RB storage

Файлы из Output Sandbox

Computing Element



RLS

Information Service

Storage Element

Job Status

submitted

waiting

ready

scheduled

running

done

Запуск задания

RB узел

Network Server

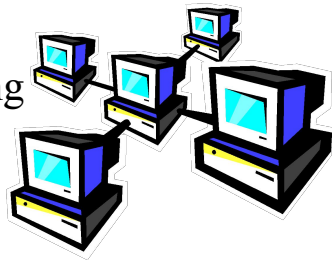
RB storage

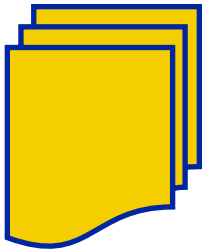
Workload Manager

edg-job-get-output <job-id>

-
CondorC

Computing Element





RLS

Information Service

Storage Element

Job Status

submitted

waiting

ready

scheduled

running

done

Запуск задания

Файлы из
Output Sandbox

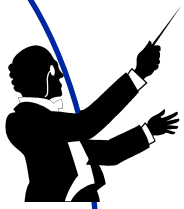
RB узел

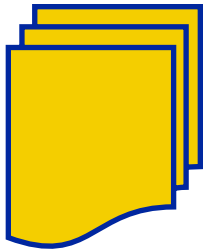
Network Server

Workload Manager

Job Contr. - CondorC

RB storage

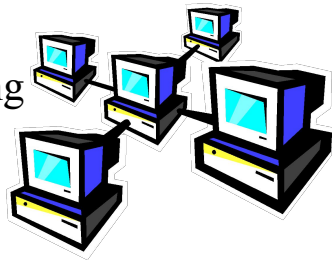




RLS

Information Service

Computing Element



Job Status

submitted

waiting

ready

scheduled

running

done

cleared

Практическая часть

Создание proxy сертификата

Перед началом работы создаём proxy сертификат

grid-proxy-init

При этом нужно будет ввести пароль, которым защищён сертификат. По умолчанию время жизни такого сертификата равно 12 часам.

```
> grid-proxy-init
Your identity: /C=RU/O=RDIG/OU=users/OU=pnpi.nw.ru/CN=Elena Martinova
Enter GRID pass phrase for this identity:
Creating proxy ..................................................... Done
Your proxy is valid until: Sat Nov 11 01:53:23 2006
```

Для получения информации о proxy-сертификате можно воспользоваться следующей командой:

grid-proxy-info –all

```
> grid-proxy-info -all
subject  : /C=RU/O=RDIG/OU=users/OU=pnpi.nw.ru/CN=Elena Martinova/CN=proxy
issuer   : /C=RU/O=RDIG/OU=users/OU=pnpi.nw.ru/CN=Elena Martinova
identity : /C=RU/O=RDIG/OU=users/OU=pnpi.nw.ru/CN=Elena Martinova
type     : full legacy globus proxy
strength : 512 bits
path     : /tmp/x509up_u10032
timeleft : 11:59:21
```

Удаление proxy сертификата

- **По окончании сеанса работы рекомендуется уничтожить proxy сертификат**

  grid-proxy-destroy

Пример задания 1

- Есть программа на PERL tstp.pl, которая использует входной файл с колонкой чисел. Каждое число возводится в квадрат и записывается в выходной файл.
- Имя входного файла передается как параметр программы. Имя выходного файла то же как и входного, но с расширением ‘out’.

Пример задания 1 (PERL файл)

- **PERL файл:**

```
#!/usr/bin/perl -w
my $inpf=$ARGV[0];
$inpf=~/(.*)\./;
my $outf=$1.'.out';

open(INP,"$inpf"); open(OUT,">$outf");
while (my $s=<INP>){
 $s*=$s; print(OUT "$s\n");
};
close(INP); close(OUT);
```

Пример задания 1 (JDL файл)

- JDL файл:

```
VirtualOrganisation = "nw_ru";
Executable = “tstp.pl";
Arguments = “tstp.inp";
StdOutput = "std.out";
StdError = "std.err";
InputSandbox={“tstp.pl",“tstp.inp"};
OutputSandbox = {"std.out", "std.err",“tstp.out"};
RetryCount = 3;
```

Пример задания 1 (INP файл)

- **INP файл:**

1

2

3

4

5

6

Просмотр доступных ресурсов

edg-job-list-match --vo <*VO*> <*job.jdl*>

- список доступных ресурсов, удовлетворяющих требованиям задания
- match making без реального запуска задания

```
>edg-job-list-match tstp.jdl

Selected Virtual Organisation name (from JDL): nw_ru
Connecting to host cluster.pnpi.nw.ru, port 7772

**********************************************************************
                COMPUTING ELEMENT IDs LIST
 The following CE(s) matching your job requirements have been found:

             *CEId*
 grid.scc.ioffe.ru:2119/jobmanager-pbs-nw_ru
 grid.spiiras.nw.ru:2119/jobmanager-pbs-nw_ru
 cluster.pnpi.nw.ru:2119/jobmanager-pbs-nw_ru
**********************************************************************
```

Запуск задания

edg-job-submit [-r *<res_id>*] [--vo *<VO>*] [-o *<output file>*] *<job.jdl>*

- **-r задание будет послано на определённый CE, идентифицируемый как <res_id>**
- **--vo название Виртуальной Организации (если не определено ранее в конфигурационном файле или JDL файле)**
- **-o идентификатор задания (jobid) будет сохранён в файле *<output file>***

  Полезно для дальнейших команд, например:

  `edg-job-status -i <input file>` (или `jobid`)

  -i означает, что jobid содержится в файле <input file>

Запуск задания 1

```
>edg-job-submit -o myid tstp.jdl

Selected Virtual Organisation name (from JDL): nw_ru
Connecting to host cluster.pnpi.nw.ru, port 7772
Logging to host cluster.pnpi.nw.ru, port 9002

================== edg-job-submit Success =====================================
 The job has been successfully submitted to the Network Server.
 Use edg-job-status command to check job current status. Your job identifier (edg_jobId) is:

- https://cluster.pnpi.nw.ru:9000/1t1_M2yCcm1FZu2XEbzEzg

 The edg_jobId has been saved in the following file:
 /home/elm/tut/task1/myid
================================================================================
```

Просмотр статуса задания

edg-job-status [-i *<input file>*] *<jobid>*

-i идентификатор задания (jobid) будет браться из файла *<intput file>*

```
>edg-job-status -i myid
**********************************************************
BOOKKEEPING INFORMATION:
Status info for the Job : https://cluster.pnpi.nw.ru:9000/1t1_M2yCcm1FZu2XEbzEzg
Current Status:     Done (Success)
Exit code:          0
Status Reason:      Job terminated successfully
Destination:        grid.scc.ioffe.ru:2119/jobmanager-pbs-nw_ru
reached on:         Fri Nov 10 14:01:14 2006
**********************************************************
```

Получение выходных файлов

Когда задание завершилось (статус Done), файлы, указанные в атрибуте OutputSandbox могут быть переданы на UI, с которого было запущено задание.

edg-job-get-output [--dir <directory>] [-i *<input file>*] *<jobid>*

-i идентификатор задания (jobid) будет браться из файла *<intput file>*

--dir файлы из OutputSandbox будут сохранены в директории <directory>

```
>edg-job-get-output --dir . -i myid
Retrieving files from host: cluster.pnpi.nw.ru ( for https://cluster.pnpi.nw.ru:9000/1t1_M2yCcm1FZu2XEbzEzg )
************************************************************************************

                JOB GET OUTPUT OUTCOME

 Output sandbox files for the job:
 - https://cluster.pnpi.nw.ru:9000/1t1_M2yCcm1FZu2XEbzEzg
 have been successfully retrieved and stored in the directory:
 /home/elm/tut/task1/elm_1t1_M2yCcm1FZu2XEbzEzg

************************************************************************************
```

Завершение задания

edg-job-cancel [-i *<input file>] <jobid>*

>edg-job-cancel -i myid

Are you sure you want to remove specified job(s)? [y/n]n :y

========== edg-job-cancel Success ==========

The cancellation request has been successfully submitted for the following job(s):

- https://cluster.pnpi.nw.ru:9000/iOomRLK6M4s_BcSm_prgRw

==========

Результат

После завершения работы команды edg-get-output в текущей директории пользователя будет создана директория, в которую скопируются 3 файла.

- std.err
- std.out
- tstp.out

Содержимое выходного файла с результатами счёта:

```
$ cat tstp.out
1
4
9
16
25
36
```

Пример задания 2

- Есть файл с исходным текстом программы на языке C ctst.c, которая выводит строку “Hello world” на стандартный вывод.
- Есть Makefile для сборки этой программы.
- Необходимо обеспечит сборку и запуск программы на удаленном ресурсе.

Пример задания 2 (JDL файл)

- JDL файл:

```
JobType="Normal";
VirtualOrganisation = "nw_ru";
Executable = "startC.sh";
StdOutput = "ctst.out";
StdError = "ctst.err";
OutputSandbox = {"ctst.out","ctst.err"};
InputSandbox = {"startC.sh","ctst.c","Makefile"};
RetryCount=3;
```

Пример задания 2 (скрипт)

- Стартовый скрипт startC.sh:

```
#!/bin/bash
make                //сборка программы
chmod +x ctst       //разрешаем запускать ее
./ctst              //запускаем
exit 0
```

Пример задания 2 (C файл)

- Текст программы на C - ctst.c:

```
#include <stdio.h>
int main(int argc, char **argv)
{
  char *name = argv[1];
  printf("\n\n\n");
  printf("Hello world!\n");
  printf("\n\n\n");
//  exit(0);
}
```

Пример задания 2 (Makefile)

- **Makefile**

```
ctst: ctst.o
        g++ ctst.o -o ctst -lm
clean:
        rm ctst.o ctst
```

Пример задания 3

- Есть файл со скриптом, который генерит набор JDL файлов, количество которых определяется входным параметром
- Затем для каждого задания случайным образом выбираются слова из системного словаря /usr/share/dict/words, которые передаются, как аргументы для каждого из запускаемых заданий.
- Все задания запускаются и контролируется процесс их выполнения.
- После завершения всех заданий (успешного или нет) – выводится результат.

Пример задания 3

- Файл echoword.sh

```
#!/bin/bash
echo "Word $1 is $2";
```

- Запуск задания

```
./submit-dictionary-jobs.sh 3
```