

# Основы работы в среде Grid:

---

**от сертификации до запуска  
заданий составных типов**

***В.Н. Ларин***

*Протвино, ИФВЭ, 03.12.2009*

# Введение

Предлагаемый материал носит *обзорный* характер и поэтому не претендует на *полноту* и *глубину*.

*Цель* лекции - дать общее представление о работе в среде **Grid** и некоторых возможностях, которые предоставляются этой средой пользователю.

Материал предназначен для пользователей, имеющих *заинтересованность* в применении **Grid**-технологий, но еще *не* знакомых или *мало* знакомых с этой средой.

# Используемые сокращения:

- CE** – **C**omputing **E**lement
- CLI** – **C**ommand **L**ine **I**nterface
- EGEE** – **E**nabling **G**rids for **E**-science**E**
- JDL** – **J**ob **D**escription **L**anguage
- LCG** – **L**HC **C**omputing **G**rid
- RDIG** – **R**ussian **D**ata **I**ntensive **G**rid
- SE** – **S**torage **E**lement
- UI** – **U**ser **I**nterface
- VO** – **V**irtual **O**rganization
- WMS** – **W**orkload **M**anagement **S**ystem
- WN** – **W**orker **N**ode

# Вопросы для обсуждения

- С чего начать?
- Как запустить задание?
- Как отслеживать его выполнение?
- Как получить результат?
- Какие типы заданий реализуются?

# С чего начать?

- Получить цифровой *сертификат пользователя*.
- Получить доступ (*account*) к **UI (User Interface)**.
- Зарегистрироваться в одной или нескольких **ВО (Виртуальных Организациях)**.

# Как получить сертификат?

- Получить доступ к *Linux*-машине или установить ОС *Linux* на своем компьютере.
- Найти информацию о получении сертификата:
  - на сайте Вашей организации (если она *участница* какого-либо *Grid-проекта*);  
в ИФВЭ – <http://www.ihep.ru/egee/> ;
  - на сайте Центра Выдачи Сертификатов  
<http://ca.grid.kiae.ru/RDIG/certificates/obtain.html>

# Страница "EGEE в ИФВЭ"

EGEE Home | Intranet Home | Search | EDMS Documents | People | Calendar | Agenda maker | Glossary

**EGEE**  
Enabling Grids  
for E-science

**RDIG**  
Russian Data  
Intensive Grid

Информация	Пресс-релизы	События	FAQs
------------	--------------	---------	------

**Информация RDIG**

- СИС. Центр базовых ГРИД сервисов
- РОС. Региональный операционный центр
- Центр выдачи сертификатов
- Регистрация ресурсного центра
- Как получить пользовательский сертификат сотрудникам ИФВЭ**
- Ресурсные центры
- Мониторинг

**Информация ИФВЭ**

- Рабочее совещание 17-19 Января 2005 г.
- О проекте
- Участники проекта
- Ссылки
- Пресс-релиз
- Фотографии
- Курсы
- Активности
- Как проехать в Протвино

## КАК ПОЛУЧИТЬ ПОЛЬЗОВАТЕЛЬСКИЙ СЕРТИФИКАТ ДЛЯ РАБОТЫ В ГРИДЕ

Центр выдачи сертификатов находится сейчас в Курчатовском институте.

Подробное описание как получить сертификат, можно прочитать на странице <http://ca.grid.kiae.ru/RDIG/certificates/obtain.html>

Каждый пользователь, согласно этой инструкции, должен сделать следующее:

- заполнить web-форму (по ссылке, приведенной выше), распечатать и принести лично одному из Registration Authority в ИФВЭ (на ваш выбор)
- скачать на свою (или другую, например, ui0001.m45.ihep.su) linux-машину скрипт, предложенный на этом же сайте, и исполнить его, чтобы получить запрос на сертификат и файл-ключ (userkey.pem)
- отослать файл с запросом на сертификат в Курчатовский институт
- сохранить файл-ключ до получения ответного письма с сертификатом на Ваш почтовый адрес

Для ИФВЭ зарегистрировано 3 Registration Authority

- Уразметов Василь Фанусович
- Бережная Александра Яковлевна
- Котляр Виктор Витальевич

08/16/2023

Основы работы в  
среде GRID

7

## Получение нового сертификата

RDIG CA выпускает сертификаты трёх типов: пользовательские, сертификаты узлов и сертификаты сервисов. Для получения любого из этих сертификатов вам необходимо сделать следующее:

1. Внимательно ознакомиться с [текущей политикой выдачи](#) сертификатов.
2. Заполнить электронную форму для получения сертификата. Ссылки на формы для различных типов сертификатов находятся в конце документа.
3. Следуя инструкциям, изложенным в конце процесса заполнения электронной формы, получить файл с запросом на сертификат, бумажную форму запроса и модуль вашего открытого ключа.
4. Следуя тем же инструкциям, отправить файл запроса в RDIG CA, дождаться подтверждения получения вашего запроса, заполнить и подписать бумажную форму.
5. Обратиться с бумажной формой запроса и удостоверением личности к вашему Registration Authority ([список всех RA](#)), который подтвердит ваши персональные данные, принадлежность к вашей организации и правильность вашего запроса.
6. После получения подтверждения от Registration Authority ваш сертификат будет подписан и выслан *на указанный вами электронный адрес* в течении трёх рабочих дней.

### Я хочу получить

- [пользовательский сертификат](#) (user certificate), [не-SSL форма](#),
- [сертификат узла](#) (host certificate), [не-SSL форма](#),
- [сертификат сервиса](#) (service certificate), [не-SSL форма](#).

Примечание: ссылкой «не-SSL форма» предполагается пользоваться тогда, когда не работает обычная HTTPS-ссылка. Но первым делом экспортируйте в браузер [корневой сертификат RDIG CA](#).





сертификаты — [запросить \(PEM\)](#) — [удалить сертификат](#) — [SSL \(PEM\)](#)

пользователи — [получить сертификат](#) — [обновить сертификат](#) — [список RA](#)

информация — [получить справку](#) — [работа с сертификатами](#) — [запросы RA](#) — [FAQ](#)

## Получение нового пользовательского сертификата

Пожалуйста, заполните все поля формы для загрузки файла сценария, который поможет вам создать пару ключей и запрос на сертификацию. После правильного заполнения всех полей запроса нажмите кнопку "Далее" и сохраните полученный файл. Для запуска полученного файла вам будет нужна совместимая с UNIX рабочая среда и работающий пакет OpenSSL.

Бумажная форма запроса будет автоматически сгенерирована на основании введенных вами данных. Пожалуйста, не забудьте распечатать и дозаполнить бумажный запрос.

**Обратите внимание:** сертификат может быть подписан только в случае личного контакта владельца сертификата, персональные данные которого будут указаны ниже, с одним из RDIG CA Registration Authorities ([список RA](#)). При этом пользователь должен иметь при себе удостоверение личности и заполненную бумажную форму.

Персональные данные владельца сертификата	
<b>Имя:</b> ваше имя (по-английски), например Ivan	<input type="text"/>
<b>Фамилия:</b> ваша фамилия (по-английски), например Petrov	<input type="text"/>
<b>E-mail:</b> адрес электронной почты, на который будут приходить все сообщения о вашем сертификате. Например vania@ru.net	<input type="text"/>
<b>Контактный телефон:</b> телефон (с кодом города), по которому с вами можно связаться. Например +7 (495) 123-45-67	<input type="text"/>
Параметры сертификата	
<b>Common Name:</b> ваши имя и фамилия (по-английски), например Ivan Petrov	<input type="text"/>
<b>Организация:</b> название организации, в которой вы работаете. Если названия вашей организации нет в списке, то вам сначала необходимо завести у себя Registration Authority. Как это сделать, написано в <a href="#">этой инструкции</a> .	<input type="text"/> <ul style="list-style-type: none"> <li>RRC KI, grid.kiae.ru</li> <li>SINP MSU, sinp.msu.ru</li> <li>JINR, jinr.ru</li> <li>PNPI, pnpi.nw.ru</li> <li>IHEP, ihep.su</li> <li>ITDP, itdp.ru</li> </ul>

# Заполните форму и следуйте инструкции

- Заполнив электронную форму на странице, показанной выше, Вы получите подробную **инструкцию** по созданию запроса на получение сертификата.
- Следуя инструкции, сгенерируйте и отправьте в **Центр Выдачи Сертификатов** запрос, распечатайте, заполните бумажную форму, подпишите ее у **Уполномоченного по регистрации (RA – Registration Authority)**.
- В процессе генерации запроса создается Ваша **ключевая пара: открытый и закрытый ключи**.
- Через **3 дня (или недели)** Вы получите из ЦВС подписанный сертификат.

# Результат сертификации

Результатом успешной сертификации являются два файла, помещенные в каталог *.globus*, созданный автоматически при генерации запроса в Вашем домашнем каталоге:

- Файл с **открытым ключом** (*public key*), куда Вы скопируете подписанный сертификат:

*usercert.pem*

- Файл, содержащий **закрытый ключ** (*private key*):

*userkey.pem*

# Виртуальная организация

**Виртуальная организация (VO)** – это объединение пользователей и ресурсов Grid для решения задач в конкретной области научных исследований в соответствии с установленными для данной VO правилами.

**Правила** регулируют доступ к вычислительным ресурсам, программному обеспечению и данным.

# Информация о VOs

- Информацию о VO, действующих в рамках проектов *EGEE* и *LCG* можно найти здесь

[https://lcg-registrar.cern.ch/virtual\\_organization.html](https://lcg-registrar.cern.ch/virtual_organization.html)

- Для доступа к ресурсам только *Российского Грид-сегмента (РГС)* можно зарегистрироваться в одной из VO, поддерживаемых *RDIG*.

Информацию о них можно найти на странице

[http://rdig-registrar.sinp.msu.ru/virtual\\_organization.html](http://rdig-registrar.sinp.msu.ru/virtual_organization.html)

# О регистрации в VO

- Познакомиться с *порядком регистрации* и *зарегистрироваться* можно на указанных выше сайтах.
- Для этого в Ваш браузер необходимо *загрузить персональный сертификат* в формате **PKCS12**.
- Для *конвертации* цифрового сертификата в этот формат в подкаталоге *.globus* нужно выполнить команду:

---

```
[larin@larin .globus]$openssl pkcs12 -export -inkey userkey.pem -in usercert.pem -out cert.p12 -name "MyCertificate"
```

# Как запустить задание?

- Войти на UI.
- Получить *proxy*-сертификат.
- Создать *JDL*-файл с описанием задания.
- Запустить задание с помощью команды

*glite-wms-job-submit\**

---

*\*Здесь и далее используются команды системы управления заданием gLite WMS via WMPProxy. Команды LCG-2 WMS и gLite WMS via NS (Network Server) обсуждаются, например, в “gLite 3 User Guide”.*

# Доступ на UI

Получив *account* на одном из доступных UI на кластере (у *администратора* кластера), осуществить вход (например, на *ui.ngc6475*):

---

```
[larin@larin ~]$ ssh larinvn@ui.ngc6475.ihep.su
larinvn@ui.ngc6475.ihep.su's password: *****
ssh(6087) Warning: Remote host denied X11 forwarding.
Last login: Thu Jun 18 15:39:46 2009 from larin.ihep.su
[larinvn@ui ~]
```



# Предварительные действия (1)

## 1. Идентификация для работы в среде GRID:

- Создать директорию *.globus* (при первом входе).
- Скопировать в нее два файла, сгенерированные при получении сертификата: *usercert.pem*, *userkey.pem*.

---

```
[larinvn@ui ~]$ ls -l ~/ .globus
```

```
total 12
```

```
-r--r--r-- 1 larinvn larinvn 5669 Oct 20 14:37 usercert.pem
```

```
-r----- 1 larinvn larinvn 963 Oct 20 14:38 userkey.pem
```

---

**Внимание!** Эти файлы необходимо *обновлять* после каждого обновления сертификата (1 раз в год).

# Предварительные действия (2)

## *2. Создание прокси-сертификата для получения доступа к ресурсам GRID.*

- Предполагается, что Вы уже член VO (например, **edu**).
- Тогда прокси-сертификат создается командой

***voms-proxy-init -voms edu***

---

По-умолчанию прокси-сертификат создается на **12 часов**. Если необходимость в нем отпала раньше, его следует **аннулировать**:

***voms-proxy-destroy***

# Пример 1. (Proxy)

```
[larinvn@ui ~]$ voms-proxy-init -voms edu
Cannot find file or dir: /home/larinvn/.glite/vomses
Enter GRID pass phrase: *****
Your identity: /C=RU/O=RDIG/OU=users/OU=ihep.su/CN=Vladislav
Larin
Creating temporary proxy ..... Done
Contacting vps102.jinr.ru:15000
[/O=Grid/OU=GlobusTest/OU=GridStudy/CN=host/vps102.jinr.ru]
"edu" Done
Creating proxy ..... Done
Your proxy is valid until Tue Jun 23 21:33:32 2009
```

# Proxu для “длинных” задач

- “Стандартный” *proxu* предоставляет не более **12 часов** для выполнения заданий.
- Возможна “пролонгация” *proxu* сервером автоматического обновления сертификатов (*myproxu*).
- Для этого нужно зарегистрировать *proxu*-сертификат на сервере с помощью команды

***myproxu-init -s <сервер> -t <время регистрации>***

- По умолчанию адрес *myproxu*-сервера берется из переменной окружения `MYPROXY_SERVER`, а время регистрации по умолчанию составляет **168 часов (7 дней)**.

# Предварительные действия (3)

## 3. Явное делегирование полномочий (рекомендуемое, но необязательное действие).

*WMProxy*-сервис взаимодействует с *WMS* от имени пользователя, поэтому последний должен делегировать ему свои полномочия. Существует два способа делегирования – автоматическое и явное. Первый способ реализуется в командах с помощью опции **-a**. Второй – с помощью опции **--delegationid (-d)** с указанием идентификатора делегирования, который определяется командой ***glite-wms-job-delegate-proxy -d <userdelegID>***

## Пример 2. (Делегирование)

```
[larinvn@ui ~]$ glite-wms-job-delegate-proxy -d $USER
```

```
Connecting to the service
```

```
https://ce.ngc6475.ihep.su:7443/glite_wms_wmproxy_server
```

```
===== glite-wms-job-delegate-proxy Success =====
```

```
Your proxy has been successfully delegated to the WMPProxy:
```

```
https://ce.ngc6475.ihep.su:7443/glite_wms_wmproxy_server
```

```
with the delegation identifier: larinvn
```

```
=====
```

Здесь **larinvn** – *идентификатор делегирования.*

# Предварительные действия (4)

## 4. Описание задания (*JDL*-файл)

Файл с описанием задания – это текстовый файл на языке *JDL* (*Job Description Language*), содержащий строки в виде пар:

*attribute = “expression”;*

Атрибуты в основном определяют:

- *тип задания;*
- *используемые файлы (“входные” и “выходные”);*
- *требования, предъявляемые к вычислительным ресурсам.*

## Пример 3. (Простейшие JDL-файлы)

```
[larinvn@ui ~]$ cat hello.jdl
```

```
Type = "Job";  
JobType = "Normal";  
Executable = "/bin/echo";  
Arguments = "Hello IHEP";  
StdOutput = "std.out";  
StdError = "std.err";  
OutputSandbox = {"std.out",  
"std.err"};
```

```
[larinvn@ui ~]$ cat script.jdl
```

```
Executable = "script.sh";  
Arguments = "infile";  
StdOutput = "script.out";  
StdError = "script.err";  
OutputSandbox =  
{"script.out", "script.err"};  
InputSandbox =  
{"/home/larinvn/script.sh",  
"/home/larinvn/infile"};
```



# Список ресурсов, доступных для запуска задания

Прежде чем запускать задание, полезно проверить какие **Вычислительные Элементы (CE)** доступны для его выполнения. Это реализуется командой:

```
glite-wms-job-list-match -a hello.jdl
```

с автоматическим делегированием, или

```
glite-wms-job-list-match -d $USER script.jdl
```

с явным делегированием.

Кроме того, эта команда позволяет **проверить синтаксис *JDL*-файла**. Однако применима только для простых заданий.

# Пример списка ресурсов

```
[larinvn@ui ~]$ glite-wms-job-list-match -a --rank hello.jdl
Connecting to the service
  https://ce.ngc6475.ihep.su:7443/glite_wms_wmproxy_server
=====
                COMPUTING ELEMENT IDs LIST
The following CE(s) matching your job requirements have been found:
                *CEId*                                *Rank*
- vps117.jinr.ru:2119/jobmanager-pbs-edu                3
- ce.ngc6475.ihep.su:2119/jobmanager-lcgpbs-edu         2
- vps105.jinr.ru:2119/jobmanager-lcgpbs-edu             2
- vps107.jinr.ru:2119/jobmanager-lcgpbs-edu            2
=====
```

# Запуск задания

На примере простейшего задания обсудим команды CLI (*Command Line Interface*), доступные пользователю при его выполнении.

---

## 1. Команда запуска задания:

```
glite-wms-job-submit -a -o jobid hello.jdl
```

- a - автоматическое делегирование полномочий *WMPoxy*;
- o - направляет в файл **jobid** идентификатор задания вида

<https://ce.ngc6475.ihep.su:9000/F2I4GacjsnO84wxftqNmog>

(Эта опция позволяет в дальнейшем указывать короткое имя файла, а не сам громоздкий идентификатор.)

# Результат выполнения команды

```
[larinvn@ui ~]$ glite-wms-job-submit -a -o jobid hello.jdl
Connecting to the service https://ce.ngc6475.ihep.su:7443/
glite_wms_wmproxy_server
===== glite-wms-job-submit Success =====
The job has been successfully submitted to the WMPProxy
Your job identifier is:
https://ce.ngc6475.ihep.su:9000/iHaJCTcsQEVd66z_15_N1w
The job identifier has been saved in the following file:
/home/larinvn/jobid
=====
```

# Статус (состояние) задания

## 2. Команда, показывающая текущий статус задания:

- с файлом, содержащим идентификатор задания  
**glite-wms-job-status -i jobid**
- с идентификатором задания  
**glite-wms-job-status [https://ce.ngc6475.ihep.su:9000/iHaJCTcsQEVd66z\\_15\\_N1w](https://ce.ngc6475.ihep.su:9000/iHaJCTcsQEVd66z_15_N1w)**

Удобство первого варианта очевидно!

# Результат выполнения команды (1)

```
[larinvn@ui ~]$ glite-wms-job-status -i jobid
```

```
*****
```

## BOOKKEEPING INFORMATION:

```
Status info for the Job: https://ce.ngc6475.ihep.su:9000/iHaJCTcsQEVd66z\_15\_N1w
```

```
Current Status: Running
```

```
Status Reason: Job successfully submitted to Globus
```

```
Destination: ce.ngc6475.ihep.su:2119/jobmanager-lcgpbs-edu
```

```
Submitted: Mon Jun 15 14:19:29 2009 MSD
```

```
*****
```

# Результат выполнения команды (2)

```
[larinvn@ui ~]$ glite-wms-job-status -i jobid
```

```
*****
```

## BOOKKEEPING INFORMATION:

Status info for the Job: [https://ce.ngc6475.ihep.su:9000/iHaJCTcsQEVd66z\\_15\\_N1w](https://ce.ngc6475.ihep.su:9000/iHaJCTcsQEVd66z_15_N1w)

Current Status: **Done (Success)**

Logged Reason(s):

- **Job terminated successfully**

Exit code: **0**

Status Reason: **Job terminated successfully**

Destination: **ce.ngc6475.ihep.su:2119/jobmanager-lcgpbs-edu**

Submitted: **Mon Jun 15 14:19:29 2009 MSD**

```
*****
```

# Все состояния заданий

- **Submitted** – задание отправлено пользователем, но не обработано *WMProxy*;
- **Waiting** – задание принято *WMProxy*, но не обработано *Workload Manager*;
- **Ready** – задание приписано к *CE*, но не передано на него;
- **Scheduled** – задание ожидает в очереди *CE*;
- **Running** – задание выполняется;
- **Done** – задание выполнилось;
- **Aborted** – задание удалено системой (*WMS*);
- **Canceled** – задание снято пользователем;
- **Cleared** – результат из *OutputSandbox* передан на *UI*.



# Получение результатов выполнения задания

## 3. Команда получения результатов выполнения задания:

```
glite-wms-job-output -i jobid
```

В этом случае все результирующие файлы помещаются в каталог */tmp/username\_<jobID>*. Однако удобнее получать результат в каталоге, определяемом с помощью опции *--dir:*

```
glite-wms-job-output --dir path_name -i jobid
```

# Результат выполнения команды (1)

```
[larinvn@ui ~]$ glite-wms-job-output -i jobid
```

```
Connecting to the service https://ce.ngc6475.ihep.su:7443/  
glite_wms_wmproxy_server
```

```
=====
```

## JOB GET OUTPUT OUTCOME

```
Output sandbox files for the job:
```

```
https://ce.ngc6475.ihep.su:9000/iHaJCTcsQEVd66z_15_N1w
```

```
have been successfully retrieved and stored in the directory:
```

```
/tmp/larinvn_iHaJCTcsQEVd66z_15_N1w
```

```
=====
```

```
[larinvn@ui ~]$ cat /tmp/larinvn_iHaJCTcsQEVd66z_15_N1w/std.out  
Hello IHEP
```

# Результат выполнения команды (2)

```
[larinvn@ui ~]$ glite-wms-job-output --dir /home/larinvn/JobOutput  
-i jobid
```

```
Connecting to the service https://ce.ngc6475.ihep.su:7443/  
glite_wms_wmproxy_server
```

```
=====
```

```
JOB GET OUTPUT OUTCOME
```

```
Output sandbox files for the job:
```

```
https://ce.ngc6475.ihep.su:9000/deMiBJ29G-0xXuxRKHwGPw
```

```
have been successfully retrieved and stored in the directory:
```

```
/home/larinvn/JobOutput
```

```
=====
```

```
[larinvn@ui ~]$ cat JobOutput/std.out
```

```
Hello IHEP
```

# Снятие задания

## 3. Команда снятия задания с выполнения:

**glite-wms-job-cancel <jobID>**

Эта команда прежде, чем снять задание, запрашивает у пользователя **подтверждение на выполнение** данной операции. Если подтверждение получено, то задание снимается с соответствующим сообщением.

# Пример выполнения команды

```
[larinvn@ui ~]$ glite-wms-job-cancel -i jobid
```

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

```
Connecting to the service
```

```
https://ce.ngc6475.ihep.su:7443/glite\_wms\_wmproxy\_server
```

```
===== glite-wms-job-cancel Success =====
```

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

```
- https://ce.ngc6475.ihep.su:9000/d5vlsZc6LbHzKleaPz3MFg
```

```
=====
```

# Типы заданий

- WMS поддерживает работу с простыми и составными заданиями.
- Тип задания описывается в *JDL*-файле двумя атрибутами: *Type* и *JobType*.
- Атрибут *Type* (*Type* = “*jobtype*”;) используется для различения простых и составных заданий и может принимать три значения:
  - ◆ “**Job**” – для простого задания (может быть опущен);
  - ◆ “**collection**” – для составного задания, в котором должен быть выполнен ряд простых заданий;
  - ◆ “**dag**” – (**direct acyclic graph**) для составного задания, в котором ряд простых заданий должен быть выполнен в определенной последовательности.

# “Подтипы” простого задания

- Атрибут *JobType* задает “подтип” (*JobType* = “*subtype*”;) простого задания и может принимать следующие значения:
  - ◆ “**normal**” – обыкновенное задание (атрибут может быть опущен);
  - ◆ “**checkpointable**” – задание с контрольными точками (сохраняет промежуточные состояния, с которых оно может быть перезапущено);
  - ◆ “**interactive**” – интерактивное задание, поддерживающее связь с точкой запуска;
  - ◆ “**mpich**” – параллельное задание, для выполнения которого требуется несколько процессоров;
  - ◆ “**Parametric**” – параметрическое задание, генерирующее несколько подзаданий из одного *JDL*-файла;
  - ◆ “**partitionable**” – сериализуемое задание, выполняющее несколько экземпляров обыкновенного задания с разными исходными данными.

# Задания типа “Коллекция”

Одной из наиболее полезных функциональных возможностей *WMProxy* является реализация запуска коллекций заданий, определенных как ряд независимых заданий. Это существенно увеличивает скорость загрузки заданий по сравнению с загрузкой отдельных заданий, а вместе с механизмом проxy-делегирования экономит процессорное время с помощью многократного использования одной и той же аутентикации (authentication) для всех заданий коллекции.



# Реализация коллекции заданий

С точки зрения пользователя существует два способа реализации заданий в виде коллекции.

- Простейший способ: использовать опцию **--collection <dirname>**, где **<dirname>** - имя каталога, в котором собраны **JDL**-файлы коллекции (*другие типы файлов не допускаются!*) в команде запуска заданий.
- Другой способ: создать один **JDL**-файл типа “*collection*” и запустить его стандартным образом.

# Пример коллекции 1-го типа

```
[larinvn@ui ~]$ ls -l Old_Tests
total 28
-rw----- 1 larinvn larinvn 114 Sep 30 15:00 glite_test.jdl
-rw----- 1 larinvn larinvn 147 Sep 30 14:59 hello.jdl
-rw----- 1 larinvn larinvn 202 Sep 30 14:58 hostname.jdl
-rw----- 1 larinvn larinvn 278 Sep 30 14:56 script.jdl
[larinvn@ui ~]$ glite-wms-job-submit -a -o coll1_id --collection Old_Tests
Connecting to the service https://ce.ngc6475.ihep.su:7443/
glite_wms_wmproxy_server
===== glite-wms-job-submit Success =====
The job has been successfully submitted to the WMPProxy
Your job identifier is:
https://ce.ngc6475.ihep.su:9000/UY9GVIYhRrbnKOugG28AIQ
The job identifier has been saved in the following file:
/home/larinvn/coll1_id
```

# Выполнение коллекции 1-го типа(1)

```
[larinvn@ui ~]$ glite-wms-job-status -i coll1_id
```

```
*****
```

## BOOKKEEPING INFORMATION:

Status info for the Job: <https://ce.ngc6475.ihep.su:9000/UY9GVIYhRrbnKOugG28AIQ>

Current Status: Done (Success)

Exit code: 0

Submitted: Wed Oct 14 13:21:10 2009 MSD

```
*****
```

## Nodes information for:

Status info for the Job: <https://ce.ngc6475.ihep.su:9000/gjGOW-zep0jn5OSdfWUVaA>

Current Status: Done (Success)

Logged Reason(s):

- Job terminated successfully

Exit code: 0

Status Reason: Job terminated successfully

Destination: ce.ngc6475.ihep.su:2119/jobmanager-lcgpbs-edu

Submitted: Wed Oct 14 13:21:10 2009 MSD

```
*****
```

# Выполнение коллекции 1-го типа(2)

Status info for the Job: <https://ce.ngc6475.ihep.su:9000/pwyKMoARSqFeTOZlSyD-xA>

Current Status: Done (Success)

Logged Reason(s):

- Job terminated successfully

Exit code: 0

Status Reason: Job terminated successfully

Destination: vps107.jinr.ru:2119/jobmanager-lcgpbs-edu

Submitted: Wed Oct 14 13:21:10 2009 MSD

\*\*\*\*\*

Status info for the Job: <https://ce.ngc6475.ihep.su:9000/sETw9SoPM-6Q9HTdgrhxA>

Current Status: Done (Success)

Logged Reason(s):

- Job terminated successfully

Exit code: 0

Status Reason: Job terminated successfully

Destination: vps107.jinr.ru:2119/jobmanager-lcgpbs-edu

Submitted: Wed Oct 14 13:21:10 2009 MSD

\*\*\*\*\*

# Выполнение коллекции 1-го типа(3)

## Status info for the Job :

<https://ce.ngc6475.ihep.su:9000/za5ZroKBEagP8QeLT5WAvQ>

Current Status: Done (Success)

Logged Reason(s):

- Job terminated successfully

Exit code: 0

Status Reason: Job terminated successfully

Destination: vps107.jinr.ru:2119/jobmanager-lcgpbs-edu

Submitted: Wed Oct 14 13:21:10 2009 MSD

\*\*\*\*\*

# Результат выполнения коллекции 1-го типа

```
[larinvn@ui ~]$ glite-wms-job-output --dir Collect1 -i coll1_id
```

```
Connecting to the service https://ce.ngc6475.ihep.su:7443/
```

```
glite_wms_wmproxy_server
```

```
=====
```

## JOB GET OUTPUT OUTCOME

```
Output sandbox files for the DAG/Collection:
```

```
https://ce.ngc6475.ihep.su:9000/UY9GVYhRrbnKOugG28AIQ
```

```
have been successfully retrieved and stored in the directory:
```

```
/home/larinvn/Collect1
```

```
=====
```

```
[larinvn@ui ~]$ ls -l Collect1
```

```
total 20
```

```
-rw-rw-r-- 1 larinvn larinvn 704 Oct 14 13:36 ids_nodes.map
```

```
drwxr-xr-x 2 larinvn larinvn 4096 Oct 14 13:36 Node_glite_test_jdl
```

```
drwxr-xr-x 2 larinvn larinvn 4096 Oct 14 13:36 Node_hello_jdl
```

```
drwxr-xr-x 2 larinvn larinvn 4096 Oct 14 13:36 Node_hostname_jdl
```

```
drwxr-xr-x 2 larinvn larinvn 4096 Oct 14 13:36 Node_script_jdl
```

# Пример коллекции 2-го типа (Type = "collection");

```
[larinvn@ui ~]$ cat mycollect.jdl
[ Type = "collection";
  InputSandbox = { "/home/larinvn /infile" };
nodes = {
  [ file = "/home/larinvn/Old_Tests/glite_test.jdl"; ],
  [ JobType = "Normal"; // hello.jdl
  Executable = "/bin/echo";
  Arguments = "Hello IHEP from collection job";
  StdOutput = "std.out";
  StdError = "std.err";
  OutputSandbox = {"std.out","std.err"}; ],
  [ file = "/home/larinvn/Old_Tests/hostname.jdl"; ],
  [ JobType = "Normal"; // script.jdl
    NodeName = "node3script";
  Executable = "script.sh";
  Arguments = "infile";
  StdOutput = "script.out";
  StdError = "script.err";
  InputSandbox = {" /home/larinvn/script.sh",root.InputSandbox};
  OutputSandbox = {" script.out"," script.err"}; ] }; ]
```

## Внимание!

В *gLite 3, Example*

6.4.1.2

“ОШИБОЧНЫЙ”

СИНТАКСИС:

Type = “Collection”;

.....

Nodes = [ ..... ];

# Запуск коллекции 2-го типа

```
[larinvn@ui ~]$ glite-wms-job-submit -a -o coll2_id mycollect.jdl
```

```
Connecting to the service https://ce.ngc6475.ihep.su:7443/  
glite_wms_wmproxy_server
```

```
===== glite-wms-job-submit Success =====
```

```
The job has been successfully submitted to the WMPProxy
```

```
Your job identifier is:
```

```
https://ce.ngc6475.ihep.su:9000/wDybQpyLqVK_DJXZZU4ZjA
```

```
The job identifier has been saved in the following file:
```

```
/home/larinvn/coll2_id
```

```
=====
```



# Результат выполнения коллекции 2-го типа

```
[larinvn@ui ~]$ glite-wms-job-output --dir Collect2 -i coll2_id
```

```
Connecting to the service https://ce.ngc6475.ihep.su:7443/
```

```
glite_wms_wmproxy_server
```

```
=====
```

## JOB GET OUTPUT OUTCOME

```
Output sandbox files for the DAG/Collection :
```

```
https://ce.ngc6475.ihep.su:9000/wDybQpyLqVK_DJXZZU4ZjA
```

```
have been successfully retrieved and stored in the directory:
```

```
/home/larinvn/Collect2
```

```
=====
```

```
[larinvn@ui ~]$ ls -l Collect2
```

```
total 20
```

```
-rw-rw-r-- 1 larinvn larinvn 660 Oct 23 15:25 ids_nodes.map
```

```
drwxr-xr-x 2 larinvn larinvn 4096 Oct 23 15:25 Node_0
```

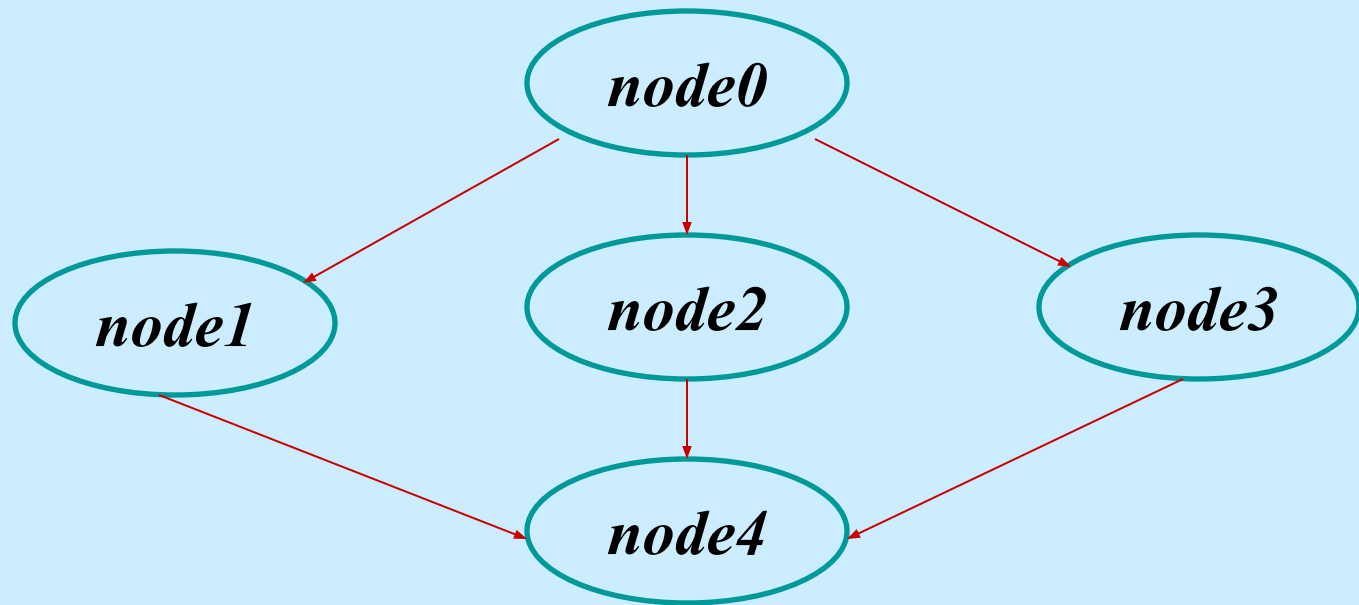
```
drwxr-xr-x 2 larinvn larinvn 4096 Oct 23 15:24 Node_1
```

```
drwxr-xr-x 2 larinvn larinvn 4096 Oct 23 15:25 Node_2
```

```
drwxr-xr-x 2 larinvn larinvn 4096 Oct 23 15:25 node3script
```

# Задание типа DAG

## Схематический пример составного DAG задания



Атрибут *dependencies* =  $\{\{node0,node1\},\{node0,node2\}, \{node0,node3\}\}$ ;  
или сокращенно: *dependencies* =  $\{ \{node1, node2, node3\}, node4 \}$  ;

# Пример задания DAG (1)

```
[larinvn@ui ~]$ cat dag1.jdl
[ Type = "dag";
  InputSandbox = { "dag1_son.sh" };           // Общий атрибут
nodes = [
  father = [ description = [
    JobType = "Normal";
    Executable = "/bin/sh";
    Arguments = "dag1_father.sh";
    InputSandbox = {"dag1_father.sh"};
    StdOutput = "father_output";
    StdError = "father_error";
    OutputSandbox = {"father_output","father_error", "son1.input", "son2.input"}; ]; ];
  son1 = [ description = [
    JobType = "Normal";
    Executable = "/bin/sh";
    InputSandbox = {root.InputSandbox, root.nodes.father.description.OutputSandbox[2]};
    Arguments = "dag1_son.sh 1";
    StdOutput = "son1.output";
    StdError = "son1.error";
    OutputSandbox = {"final1.input","son1.output","son1.error"}; ]; ];
```

# Пример задания DAG (2)

```
son2 = [ description = [
  JobType = "Normal";
  Executable = "/bin/sh";
  InputSandbox = {root.InputSandbox, root.nodes.father.description.OutputSandbox[3]};
  Arguments = "dag1_son.sh 2";
  StdOutput = "son2.output";
  StdError = "son2.error";
  OutputSandbox = {"final2.input", "son2.output", "son2.error"}; ]; ]
final = [ description = [
  JobType = "Normal";
  Executable = "/bin/sh";
  InputSandbox = {"dag1_final.sh",
  root.nodes.son1.description.OutputSandbox[0],
  root.nodes.son2.description.OutputSandbox[0]};
  Arguments = "dag1_final.sh";
  StdOutput = "dag.out";
  StdError = "dag.err";
  OutputSandbox = {"dag.out", "dag.err"}; ]; ]
dependencies = { {father, {son1, son2}}, {son1, final}, {son2, final} }; ]
```

# Результат выполнения DAG задания

Запуская и получая результат “стандартным” образом, получим **4 каталога** с именами, заданными для узлов, в которых содержатся выходные файлы, указанные в соответствующих *OutputSandbox*:

```
[larinvn@ui ~]$ ls -l ~/dag1_Output
total 12
-rw-rw-r-- 1 larinvn larinvn 0 Oct 2 13:31 father
-rw-rw-r-- 1 larinvn larinvn 16 Oct 2 13:31 son1
-rw-rw-r-- 1 larinvn larinvn 16 Oct 2 13:31 son2
-rw-rw-r-- 1 larinvn larinvn 62 Oct 2 13:31 final
```

# Параметрическое задание

**Параметрическое** задание генерирует **множество заданий** из одного **JDL-файла**.

Т.о. оно только **формально** относится к **простым** заданиям, а **фактически** является **составным**.

Применяется в тех случаях, когда нужно выполнить ряд **схожих**, но **не идентичных** заданий.

Достигается это определением одного или нескольких **атрибутов** как **параметров**. Для этого в их значениях используется **ключевое слово** **\_PARAM\_**.

# Пример параметрического задания

```
[larinvn@ui Param]$ cat param_pnpi.jdl
[JobType = "Parametric";
 Executable = "/bin/sh";
 Arguments = "message_PARAM_.sh";
 InputSandbox = "message_PARAM_.sh";
 Parameters = 6;
 ParameterStep = 2;
 ParameterStart = 0;
 StdOutput = "param_out_PARAM_.txt";
 StdError = "param_err_PARAM_.txt";
 OutputSandbox = {"param_out_PARAM_.txt","param_err_PARAM_.txt"};
]
```

# Выполнение параметри- ческого задания (1)

```
[larinvn@ui Param_pnpi]$ glite-wms-job-status -i param_id
```

```
*****
```

## BOOKKEEPING INFORMATION:

Status info for the Job: <https://ce.ngc6475.ihep.su:9000/0CEALzc74OLJMZ4u8bjcMg>

Current Status: Running

Submitted: Tue Oct 27 15:31:04 2009 MSK

```
*****
```

## - Nodes information for:

Status info for the Job:

<https://ce.ngc6475.ihep.su:9000/XT71FvR7d79TEC5nHI9PAQ>

Current Status: Running

Status Reason: Job successfully submitted to Globus

Destination: vps117.jinr.ru:2119/jobmanager-pbs-edu

Submitted: Tue Oct 27 15:31:04 2009 MSK

```
*****
```



# Выполнение параметри- ческого задания (2)

## Status info for the Job:

[https://ce.ngc6475.ihep.su:9000/\\_yogAtmDwDZsvco2bwPprA](https://ce.ngc6475.ihep.su:9000/_yogAtmDwDZsvco2bwPprA)

Current Status: Running  
Status Reason: Job successfully submitted to Globus  
Destination: vps117.jinr.ru:2119/jobmanager-pbs-edu  
Submitted: Tue Oct 27 15:31:04 2009 MSK

\*\*\*\*\*

## Status info for the Job:

<https://ce.ngc6475.ihep.su:9000/ePMp6VntdB-mtRGy3XCVBw>

Current Status: Running  
Status Reason: Job successfully submitted to Globus  
Destination: vps107.jinr.ru:2119/jobmanager-lcgpbs-edu  
Submitted: Tue Oct 27 15:31:04 2009 MSK

\*\*\*\*\*

# Результат выполнения параметрического задания

```
[larinvn@ui Param]$ ls -l Output
total 16
-rw-rw-r-- 1 larinvn larinvn 501 Oct 27 15:41 ids_nodes.map
drwxr-xr-x 2 larinvn larinvn 4096 Oct 27 15:41 Node_0
drwxr-xr-x 2 larinvn larinvn 4096 Oct 27 15:41 Node_2
drwxr-xr-x 2 larinvn larinvn 4096 Oct 27 15:41 Node_4
```

---

Здесь директориям, содержащим результаты выполнения подзаданий, присваиваются такие же имена, как и в "коллекции". Номера узлов – значения параметра.

# Коротко о других типах заданий (“checkpointable”)

- Задания с контрольной точкой

***JobType = “checkpointable”;***

Простое задание, для которого WMS поддерживает возможность периодического сохранения состояния, начиная с которого задание может быть перезапущено.

Требует дополнительных атрибутов: ***JobSteps*** (задает максимальное число контрольных точек или содержит список их имен) и ***CurrentStep*** (задает точку, с которой задание должно быть запущено).

# Коротко о других типах заданий (“interactive”)

- **Интерактивное задание**

***JobType = “interactive”;***

Простое задание, поддерживающее во время выполнения прямой контакт с “точкой запуска”. При этом стандартные потоки `stdin`, `stdout` и `stderr` перехватываются на исполнительном компьютере (*WN*) и перенаправляются на компьютер запуска, на котором должен быть стартован X-сервер, открывающий окно для ввода и вывода.

# Коротко о других типах заданий (“partitionable”)

- Сериализуемое задание

*JobType* = “*partitionable*”;

Как и параметрическое задание, только формально является простым, но при запуске генерируется **dag**-задание из нескольких независимых **серийных** заданий (однократно описанных в *JDL*-файле) и двух специальных заданий: пре- и пост-обработки, которые также описываются в *JDL*-файле:

**prejob** = [ ... ];

**postjob** = [ ... ];

# Коротко о других типах заданий (“mpich”)

- Параллельные задания

*JobType* = “*mpich*”;

*WMS* обеспечивает ограниченную поддержку многопроцессорных параллельных заданий, использующих *MPI*-протокол. В *JDL*-файле указывается атрибут *NodeNumber*, определяющий необходимое число процессоров. Кроме того, к значению атрибута *Requirements* автоматически добавляется логическое выражение, определяющее выбор *CE* с числом процессоров большим *NodeNumber* и установленной средой *MPICH*.

# Ссылки

- S. Burke, S. Campana, A.D. Peris, et al. *gLite 3 User Guide*, 2007. <https://edms.cern.ch/file/722398/1.1/gLite-3-UserGuide.pdf>
- F. Pacini. *Job Description Language Attributes Specification*, EGEE, 2007. <https://edms.cern.ch/document/590869/1>
- В.Н. Коваленко. *Комплексное программное обеспечение грида вычислительного типа*, ИПМ РАН, Москва, 2007.
- *Практикум ПИЯФ РАН*.  
<http://egee.pnpi.nw.ru/cgi/index.cgi?l1=6&l2=2>

# Заключение

The END...?!