NORDUGRID

19/8/2003

Extended Resource Specification Language

Contents

Chapter 1

Introduction

Execution of jobs at batch clusters is the major part of the data analysis, production and other computational tasks and applications in many research areas. In the Grid infrastructure [1], such clusters are not necessarily local to a job submission site, but can be widely distributed. This implies extra requirements for the proper description of job options. The Globus project [2] developed a Grid middleware toolkit, which makes use of the *Resource Specification Language (RSL)* [3] to parse job options and definitions to resource management systems. The NorduGrid project [4] developed an architectural solution [5, 6] for a Grid facility, suitable for complex tasks, like, for example, a High Energy Physics production. To match the complexity of such tasks, this solution requires certain extensions to the RSL.

To describe a task to be submitted to the NorduGrid resources, an extended version of the Globus RSL is used. The extensions concern not only introduction of new attributes, but also differentiation between two levels of the job options specifications:

> **User-side RSL**, i.e., the set of attributes specified by a user in a job-specific file. This file is interpreted by the *User Interface (UI)* [7], and after the necessary modifications is parsed to the *Grid Manager (GM)* [8]
>
> **GM-side RSL**, i.e., the set of attributes pre-processed by the UI, and ready to be interpreted by the GM

A user only has to know the user-side part, and utilize it to describe the Grid tasks. The Grid Manager, however, uses slightly different notations, supplied by the User Interface.

In what follows, the description of the NorduGrid-extended RSL, further denoted as **xRSL**, is given, using the following notations:

`<xxxx>`	parameter to be substituted with a corresponding string or a number
`[xxxx]`	optional parameter
`xxx|yyy|zzz`	list of possible values of a parameter
`-"-`	”same as above”

Chapter 2

xRSL syntax and rules

For a complete description of Globus RSL, see reference [3]. xRSL uses the same syntax conventions, although changes the meaning and interpretation of some attributes.

2.1 Syntax

A Grid task is described by the mean of xRSL attributes, which can be either parsed via a command-line, or, more conveniently, be collected a so-called xRSL-file (suggested extension *.xrsl*). Such a file contains a plain list of attribute strings and boolean operands "&" (for AND) and "—" (for OR).

Typically, an xRSL job description starts with an ampersand ("&") , to indicate implicit **conjunction** of all the attributes:

```
&(attribute1=value1)(attribute2=value2)...
```

Whenever a **disjunct**-request of two or more attributes is needed, the following construction can be used:

```
(|(attribute=value1)(attribute=value2)...)
```

In expressions, the following operands are allowed:

```
=   !=   >   <   >=   <=
```

Commented lines should start with "(*" and be closed with "*)":

```
(*attribute=value1*)
```

Multiple job description in one file is realized via a standard Globus RSL multi-request operand "+", which should precede multiple job description:

```
+(&(...))(&(...))(&(...))
```

The xRSL attributes can be written in a single string, or split in lines; blank spaces between `(attribute=value)` pairs are ignored.

2.2 URLs

File locations in NorduGrid can be specified both as local file names, and as Internet standard *Uniform Resource Locators (URL)*. However, there are some additional *options*, used by the Grid Manager.

The following transfer protocols and metadata servers are supported:

- `ftp` – ordinary *File Transfer Protocol (FTP)*
- `gsiftp` – GridFTP, the Globus-enhanced FTP
- `http` – ordinary *Hyper-Text Transfer Protocol (HTTP)*
- `https` – HTTP with Globus GSI authentication
- `ldap` – ordinary *Lightweight Data Access Protocol (LDAP)* [9]
- `rc` – Globus *Replica Catalog (RC)* [10]
- `rls` – Globus/EDG *Replica Location Service (RLS)* [11]

An URL can be used in a standard form, i.e.

```
<protocol>://host[:port]/<file>
```

Or, to enhance the performance, it can have additional options:

```
<protocol>://host[:port][;option[;option[...]]]/<file>
```

For a metadata service URL, construction is the following:

```
rc://rc://[location[|location[...]]@]<host>[:port]/<DN>/<lfn>
rls://[url[|url[...]]@]<host>[:port]/<lfn>
```

Here the URL components are:

`location`	`<location_name_in_RC>[;option[;option[...]]]`
`host[:port]`	IP address of a server
`DN`	Distinguished Name (as in LDAP) of an RC collection
`lfn`	Logical File Name
`url`	URL of the file as registered in RLS
`file`	local to the host file name with a full path

The following options are supported:

`threads=<number>` specifies number of parallel streams to be used by GridFTP; default value is 1, maximal value is 10

`cache=yes|no` indicates whether the GM should cache the file; default is `yes`

`secure=yes|no` indicates whether the GridFTP data channel should be encrypted; default is `no`

Examples of URLs are:

```
http://grid.domain.org/dir/script.sh
gsiftp://grid.domain.org:2811;threads=10/dir/input_12378.dat
ldap://grid.domain.org:389/lc=collection1,rc=Nordugrid,dc=nordugrid,dc=org
rc://grid.domain.org/lc=collection1,rc=Nordugrid,dc=nordugrid,dc=org/zebra/f1.zebra
```

Chapter 3

Attributes

Most of the attributes introduced by Globus are supported, some with modifications as indicated in this document. Several new attributes are introduced, of which some are to be specified in the user's script, and others are internal for the GM (are added or modified by the UI).

Attribute names are case-insensitive, although assigned values may well be case-sensitive, if they represent file names, environment variables etc.

3.1 User-side attributes

The following attributes can be specified in a user's xRSL script. Some are to be modified by the UI before being passed to the GM.

executable

User input: `(executable=<string>)`

GM input: `(executable=/bin/echo)`

Example: `(executable="myprog.exe")`

The executable to be submitted to LRMS.

`string` file name (including path), local to the computing element (CE)

If an executable has to be transferred from the submission node, it has to be specified in the `inputFiles` list, otherwise it will be added to that list by the UI.

If the file name starts with a leading slash ("`/`"), it is considered to be **the full path to the executable at a CE**; otherwise the location of the file is **relative** to the session directory (where job input and files are stored). If the file name starts with an environment variable ("`$...`"), the value of this variable is resolved at the remote computing element.

For internal GM use, the UI creates a dummy executable attribute with, e.g., "`/bin/echo`" as an argument (unless the specified executable itself starts with "`/`"). The actual executable is transferred by the UI to lists in the `arguments`, `inputFiles` and `executables` internal GM attributes.

arguments

User input: `(arguments=<string> [string] ... )`

GM input: `(arguments=<executable> <string> [string] ... )`

Example: `(arguments="10000" $(ATLAS)/input.dat)`

List of the arguments for the executable.

`string` an argument

`executable` the executable to be run by LRMS, taken by the UI from the user-specified `executable` attribute

inputFiles

User input: `(inputFiles=(<filename> <location>) ... )`

GM input: `(inputFiles=(<filename> <URL>)`
`(<filename> [size][.checksum]) ... )`

Example: `(inputFiles=(file1.dat gsiftp://grid.quark.lu.se/scratch/test.dat)`
`(file2.dat /scratch/bigfile.dat)`
`(file3.dat ""))`

List of files to be copied to the computing element before the execution.

`filename` file name, local to the computing element and always relative to the session directory

`location` location of the file (`gsiftp`, `https`, `ftp`, `http` URLs, or a path, local to the submission node). If void (””), the input file is taken from the submssion directory.

`URL` URL of the file (`gsiftp`, `https`, `ftp`, or `http` protocol)

`size` file size in bytes

`checksum` file checksum (as returned by `cksum`)

If the list does not contain the standard input file (as specified by `stdin`) and/or the executable file (as specified by `executable` if the given name), the UI appends these files to the list. If the `<location>` is a URL, it is passed by the UI to the GM without changes; if it is a local path (or void, “”), the UI converts it to `<size>` and `<checksum>` and uploads those files to the execution cluster.

executables

User input: `(executables=<string> [string] ...)`

GM input: `-’’-`

Example: `(executables=myscript.sh myjob.exe)`

List of files from the `inputFiles` set, which will be given executable permissions.

`string` file name, local to the computing element and relative to the session directory

If the executable file (as specified in `executable` and is relative to the session directory) is not listed, it will be added to the list by the UI.

outputFiles

User input: `(outputFiles=(<string> <URL>) ... )`

GM input: `-’’-`

Example: `(outputFiles=(file1.dat gsiftp://grid.uio.no/storage/file_num_11)`
`(file2 rc://grid.fi.uib.no/group1/result2) )`

List of files to be retrieved by the user or uploaded by the GM and registered in a Replica Catalog.

`string`	file name, local to the *Computing Element (CE)*
`URL`	URL of the remote file (`gsiftp`, `https`, `ftp`, `http` or a Replica Catalog pseudo-URL); if void (""), the file is kept for manual retrieval.

Using a RC pseudo-URL (see Section 2.2), you should make sure that the location is already defined in the RC. If few locations are specified, only those found in the RC will be used. GM will store output files in **one** location only. If the first one in the list fails, it would try the next. If no locations are specified, all found in the RC will be used.

If in RC pseudo-URL the component `host[:port]/DN` is not specified, the one given in the `replicaCollection` attribute is used.

If the list does not contain standard output and/or standard error files (as specified by `stdout/stderr`), the UI appends these file names to the list. If the `<URL>` is not specified (void, ""), files will be downloaded by the user via the UI.

cpuTime

User input:	`(cpuTime=<time>)`
GM input:	`-''-`
Example:	`(cpuTime=240)`

Maximal CPU time request for the job.

`time`	time (minutes)

If only number is specified, the time is assumed to be minutes. Otherwise, a free format is accepted, i.e., any of the following will be interpreted properly:

```
1 week
3 days
2 days, 12 hours
1 hour, 30 minutes
36 hours
9 days
240 minutes
```

memory

User input:	`(memory=<integer>)`
GM input:	`-''-`
Example:	`(memory>=500)`

Memory required for the job.

`integer`	size (Mbytes)

disk

User input:	`(disk=<integer>)`
GM input:	`-''-`
Example:	`(disk=500)`

Disk space required for the job.

`integer` disk space, Mbytes

runTimeEnvironment

User input:	`(runTimeEnvironment=<string>)`
GM input:	`-’’-`
Example:	`(runTimeEnvironment=Atlas-1.1.0)`

Required runtime environment

`string` environment name

The site to submit the job to will be chosen by the UI among those advertising specified runtime environments. Before starting the job, the GM will set up environment variables and pathes according to those requested.

To request several environments, repeat the attribute string:
`(runTimeEnvironment=ENV1)(runTimeEnvironment=ENV2)` etc.

Before being submitted to the GM, a grouping is done by the UI, e.g., `(runTimeEnvironment=ENV1 ENV2)`.

To make a disjunct-request, use a boolean expression:
`(|(runTimeEnvironment=env1)(runTimeEnvironment=env2))`.

Runtime environment string interpretation is case-insensitive. Substring specification is possible: in such a case, a partial match will be substituted with a full advertised in the MDS value and only then submitted to the GM.

Use the “>=” operator to request a version “equal or higher”.

middleware

User input:	`(middleware=<string>)`
GM input:	`-’’-`
Example:	`(middleware=NorduGrid-0.3.99)`

Required middleware.

`string` Grid middleware name.

The site to submit the job to will be chosen by the UI among those advertising specified middleware. Usage is identical to that of the `runTimeEnvironment`. Use the “>=” operator to request a version “equal or higher”. Request `(middleware=nordugrid)` defaults to `(middleware>=nordugrid-0.0.0.0)`.

stdin

User input:	`(stdin=<string>)`
GM input:	`-’’-`
Example:	`(stdin=myinput.dat)`

The standard input file.

`string` file name, local to the computing element

The standard input file should be listed in the `inputFiles` attribute; otherwise it will be forced to that list by the UI.

stdout

User input:	`(stdout=<string>)`
GM input:	`-’’-`
Example:	`(stdout=myoutput.txt)`

The standard output file.

`string` file name, local to the computing element and relative to the session directory.

The standard output file should be listed in the `outputFiles` attribute; otherwise it will be forced to that list by the UI. If the standard output is not defined, UI assigns a name.

stderr

User input:	`(stderr=<string>)`
GM input:	`-’’-`
Example:	`(stderr=myjob.err)`

The standard error file.

`string` file name, local to the computing element and relative to the session directory.

The standard error file should be listed as an `outputFiles` attribute; otherwise it will be forced to that list by the UI. If the standard error is not defined, UI assigns a name.

join

User input:	`(join=yes|no)`
GM input:	`-’’-`
Example:	`(join=yes)`

If ”yes”, joins `stderr` and `stdout` files into the `stdout` one. Default is `no`.

gmlog

User input:	`(gmlog=<string>)`
GM input:	`-’’-`
Example:	`(gmlog=myjob.log)`

The job log file, containing all the job-related messages from the GM.

`string` file name, local to the computing element and relative to the session directory

The job log file should be listed as an `outputFiles` attribute; otherwise it will be forced to that list by the UI.

jobName

User input:	`(jobName=<string>)`
GM input:	`-’’-`
Example:	`(jobName=MyJob)`

User-specified job name.

`string`	job name

This name is meant for convenience of the user. It can be used to select the job while using the UI. It is also available through the Information System.

ftpThreads

User input:	`(ftpThreads=<integer>)`
GM input:	`-’’-`
Example:	`(ftpThreads=4)`

Defines how many parallel streams will be used by the GM during `gsiftp` transfers of files.

`integer`	a number from 1 to 10

If not specified, parallelism is not used.

cluster

User input:	`(cluster=<string>)`
GM input:	`-’’-`
Example:	`(cluster=nbi)`

The name of the execution cluster.

`string`	known cluster name, or a substring of it

Use this attribute to explicitly force job submission to a cluster, or to avoid such. The job will not be submitted if the cluster does not satisfy other requirements of the job. Disjunct-requests of the kind `(|(cluster=clus1)(cluster=clus2))` are supported. To exclude a cluster, use `(cluster!=clus3)`.

queue

User input:	`(queue=<string>)`
GM input:	`-’’-`
Example:	`(queue=pclong)`

The name of the remote batch queue.

`string`	known queue name

Use this attribute to explicitly force job submission to a queue.

startTime

User input:	`(startTime=<time>)`
GM input:	`(startTime=<tttt>)`
Example:	`(startTime="2002-05-25 21:30")`

Time to start job processing.

`time`	time string, YYYY-MM-DD hh:mm:ss
`tttt`	time string, YYYYMMDDhhmmss[Z] (converted by the UI from `time`)

lifeTime

User input:	`(lifeTime=<time>)`
GM input:	`(lifeTime=<tttt>)`
Example:	`(lifeTime=60)`

Maximal time to keep job files (the session directory) on the gatekeeper upon job completion.

`time`	time (days)
`tttt`	time (seconds, converted by the UI from `time`)

Typical life time is 1 day (24 hours). Specified life time can not exceed local settings.

notify

User input:	`(notify=<string> [string] ... )`
GM input:	`-''-`
Example:	`(notify="be your.name@your.domain.com")`

Request e-mail notifications on job status change.

`string` string of the format: `[b][q][f][e][c] user1@domain1 [user2@domain2] ...`
here flags indicating the job status are:
`b` – begin (PREPARING)
`q` – queued (INLRMS)
`f` – finalizing (FINISHING)
`e` – end (FINISHED)
`c` – cancellation (CANCELLED)

No more than 3 e-mail addresses per status change accepted.

replicaCollection

User input:	`(replicaCollection=<URL>)`
GM input:	`-''-`

Example:	`(replicaCollection="ldap://grid.uio.no:389/lc=TestCollection,`
	`rc=NorduGrid,nordugrid,dc=org")`

Location of a logical collection in the Replica Catalog.

`URL`	LDAP directory specified as an URL (`ldap://host[:port]/dn`)

rerun

User input:	`(rerun=<integer>)`
GM input:	`-''-`
Example:	`(rerun=2)`

Number of reruns (if a system failure occurs).

`integer`	an integer number

If not specified, the default is 0. Default maximal allowed value is 2.

architecture

User input:	`(architecture=<string>)`
GM input:	
Example:	`(architecture=i686)`

Request a specific architecture.

`string`	architecture (e.g., as produced by uname -a)

nodeAccess

User input:	`(nodeAccess=inbound|outbound)`
GM input:	
Example:	`(nodeAccess=inbound)`

Request cluster nodes with inbound or outbound IP connectivity. If both are needed, a conjunct request should be specified.

dryRun

User input:	`(dryRun=yes|no)`
GM input:	`-’’-`
Example:	`(dryRun=yes)`

If ”yes”, do dry-run: RSL is parsed, but no job submission to LRMS is made. Should be used for xRSL validation.

rsl_substitution

User input:	`(rsl_substitution=(<string1> <string2>))`
GM input:	`-’’-`
Example:	`(rsl_substitution=(ATLAS /opt/atlas))`

Substitutes `<string2>` with `<string1>` for **internal** RSL use.

`string1`	new internal RSL variable
`string2`	any string, e.g., existing combination of variables or a path

Use this attribute to simplify xRSL editing. Only one pair per substitution is allowed. To request several substitution, concatenate such requests. Bear in mind that substitution must be defined **prior** to actual use of a new variable `string1`.

environment

User input:	`(environment=(<VAR> <string>) [(<VAR> <string>)] ... )`
GM input:	`-’’-`

Example:	`(environment=(ATLSRC /opt/atlas/src)`
	`(ALISRC /opt/alice/src))`

Defines execution shell environment variables.

`VAR`	new variable name
`string`	any string, e.g., existing combination of variables or a path

Use this to define variables at an execution site.

count

User input:	`(count=<integer>)`
GM input:	`-’’-`
Example:	`(count=4)`

Specifies amount of sub-jobs to be submitted for parallel tasks.

`integer`	a number (default is 1)

3.2 GM-side attributes

The following attributes are constructed by the UI and are parsed to the GM, although users may specify them manually (**not advised!**).

sstdin

User input:	
GM input:	`(sstdin=<filename>)`
Example:	`(sstdin=myinput.dat)`

Internal attribute for the standard input. Can also be spelled `stdinput`.

`filename`	standard input file name

action

User input:	
GM input:	`(action=request|cancel|clean)`
Example:	`(action=request)`

Action to be taken by the gatekeeper: submit the job, cancel job execution, or clear the results of the job (also cancels the job).

savestate

User input:	
GM input:	`(savestate=yes|no)`
Example:	`(savestate=yes)`

If ”yes”, input RSL is stored in a temporary file at the gatekeeper. Must be always set as ”yes” in the current implementation.

lrmstype

User input:	
GM input:	`(lrmstype=<string>)`
Example:	`(lrmstype=pbs)`

LRMS type, indicating which submission script is to be invoked.

`string`	LRMS type (at the moment, only ”pbs” is supported)

hostName

User input:	
GM input:	`(hostname=<string>)`
Example:	`(hostName=grid.quark.lu.se)`

Submission host name.

`string`	host name, as passed by the UI

jobid

User input:	
GM input:	`(jobid=<string>)`
Example:	`(jobid=grid.quark.lu.se:2119/jobmanager-ng/15711.1017133827)`

Unique job identification string, needed for cancellation and clean-up.

`string`	global job ID

It can also be provided during submission of the job and should be unique to a computing element (cluster).

3.3 Unsupported Globus RSL attributes

The following Globus attributes are not supported by the NorduGrid middleware. Whenever they are specified, a warning is issued by the UI, and the corresponding attribute is ignored.

3.3.1 Unsupported RSL 1.0 attributes

- `(resourceManagerContact=<string>)`
- `(directory=<string>)`
- `(maxCpuTime=<time>)`
- `(maxWallTime=<time>)`
- `(maxTime=<time>)`
- `(maxMemory=<memory>)`

- (minMemory=<memory>)
- (gramMyJob=independent|collective)
- (project=<string>)
- (hostCount=<number>)
- (label=<string>)
- (subjobCommsType=blocking-join|independent)
- (subjobStartType=strict-barrier|loose-barrier|no-barrier)

3.3.2 Unsupported GRAM RSL attributes

- (directory=<string>)
- (fileCleanUp=<array>)
- (fileStageIn=<array>)
- (fileStageInShared=<array>)
- (fileStageOut=<array>)
- (gassCache=<path>)
- (gramMyJob=independent|collective)
- (hostCount=<number>)
- (jobType=<string>)
- (libraryPath=<path>)
- (maxCpuTime=<time>)
- (maxWallTime=<time>)
- (maxTime=<time>)
- (maxMemory=<memory>)
- (minMemory=<memory>)
- (project=<string>)
- (remoteIoUrl=<string>)
- (scratchDir=<string>)

Appendix A

Examples

A.1 User-side xRSL script

```
&
(* test run: if "yes", only submits RSL without actual job start *)
     (dryRun=no)
(* some local variables defined for further convenience *)
     (rsl_substitution=(TOPDIR /home/oxana))
     (rsl_substitution=(NGTEST $(TOPDIR)/examples/ngtest))
     (rsl_substitution=(BIGFILE /scratch/oxana/100mb.tmp))
(* some environment variables, to be used by the job *)
     (environment=(ATLAS /opt/atlas) (CERN /cern))
(* the main executable file to be staged in and submitted to the PBS *)
     (executable=checkall.sh)
(* the arguments for the executable above *)
     (arguments=pal)
(* files to be staged in before the execution *)
     (inputFiles = (be_kaons "")
     (file1 gsiftp://grid.uio.no$(TOPDIR)/remfile.txt)
      (bigfile.dat $(BIGFILE) )    )
(* files to be given executable permissions after staging in *)
     (executables=be_kaons)
(* files to be staged out after the execution *)
     (outputFiles=
        (file1 gsiftp://grid.tsl.uu.se/tmp/file1.tmp)
        (100mb.tmp rc://grid.fi.uib.no/bigfile)
        (be_kaons.hbook  gsiftp://grid.quark.lu.se$(NGTEST)/kaons.hbook)   )
(* location of the Replica collection *)
   (replicaCollection="ldap://grid.uio.no:389/lc=TestCollection,rc=NorduGrid,dc=nordugrid,dc=org")
(* user-specified job name *)
     (jobName=NGtest)
(* standard input file *)
     (stdin="myinput.dat")
(* standard output file *)
     (stdout="myoutput.dat")
(* standard error file *)
     (stderr="myerror.dat")
(* GM logs directory name *)
     (gmlog="gmlog")
(* flag whether to merge stdout and stderr *)
     (join="no")
(* request e-mail notification on status change *)
```

```
     (notify="bqfe oxana.smirnova@quark.lu.se oxana.smirnova@cern.ch")
(* specific queue to submit the job *)
     (queue=atlas)
(* maximal CPU time required for the job, minutes for PBS*)
     (CpuTime=60)
(* maximal time for the session directory to exist on the remote node, days *)
     (lifeTime=7)
 (* memory required for the job, Mbytes *)
     (Memory=200)
(* wall time to start job processing *)
     (startTime="2002-04-28 17:15:00")
(* disk space required for the job, Mbytes *)
     (Disk=500)
(* required architecture of the execution node *)
     (architecture=i686)
(* required run-time environment *)
     (runTimeEnvironment="Atlas-1.1")
(* number of re-runs, in case of a system failure *)
     (rerun=2)
```

A.2 GM-side xRSL script

```
&
(* saves RSL in a temporary file if "yes"*)
     (savestate=yes)
(* job submission to be performed if action is "request" *)
     (action=request)
(* LRMS type, so far only pbs *)
     (lrmstype=pbs)
(* submission host name *)
     (hostName="grid.quark.lu.se")
(* test run: if "yes", only submits RSL without actual job start *)
     (dryRun=no)
(* some local variables defined for further convenience *)
     (rsl_substitution=(TOPDIR /home/oxana))
     (rsl_substitution=(NGTEST $(TOPDIR)/examples/ngtest))
     (rsl_substitution=(BIGFILE /scratch/oxana/100mb.tmp))
(* some environment variables, to be used by the job *)
     (environment=(ATLAS /opt/atlas) (CERN /cern))
(* a dummy executable *)
     (executable=/bin/echo)
(* the main executable itself, and its arguments *)
     (arguments=checkall.sh pal)
(* files to be staged in before the execution *)
     (inputFiles = (checkall.sh 210.279320915)
        (myinput.dat 14.3980640923)
        (be_kaons 880788.2035414420)
        (file1 gsiftp://grid.uio.no$(TOPDIR)/remfile.txt)
        (bigfile.dat 104857600.4087847787)
     )
(* files to be given executable permissions after staging in *)
     (executables=checkall.sh be_kaons)
(* files to be staged out after the execution *)
     (outputFiles=(file1 gsiftp://grid.tsl.uu.se/tmp/file1.tmp)
        (100mb.tmp rc://grid.fi.uib.no/bigfile)
        (be_kaons.hbook  gsiftp://grid.quark.lu.se$(NGTEST)/kaons.hbook)
        (myoutput.dat  gsiftp://grid.quark.lu.se/home/oxana/examples/ngtest/myoutput.dat)
        (myerror.dat  gsiftp://grid.quark.lu.se/home/oxana/examples/ngtest/myerror.dat)
    )
(* location of the Replica collection *)
  (replicaCollection="ldap://grid.uio.no:389/lc=TestCollection,rc=NorduGrid,dc=nordugrid,dc=org")
(* user-specified job name *)
     (jobName=NGtest)
(* standard input file *)
     (stdin="myinput.dat")
(* standard output file *)
     (stdout="myoutput.dat")
(* standard error file *)
     (stderr="myerror.dat")
(* flag whether to merge stdout and stderr *)
     (join="no")
(* request e-mail notification on status change *)
     (notify="bqfe oxana.smirnova@quark.lu.se oxana.smirnova@cern.ch")
(* specific queue to submit the job *)
     (queue=pc)
(* CPU time required for the job, minutes for PBS*)
     (CpuTime=60)
(* maximal time for the session directory to exist on the remote node, seconds *)
     (lifeTime=604800)
(* memory required for the job, Mbytes *)
     Memory=200)
(* wall time to start job processing *)
     (startTime=20020128171500)
(* disk space required for the job, Mbytes *)
     (Disk=500)
(* required architecture of the execution node *)
     (architecture=i686)
(* required run-time environment *)
     (runTimeEnvironment="Atlas-1.1")
(* number of re-runs, in case of a system failure *)
     (rerun=2)
(* job ID - needed for cancellation and cleanup *)
     (*(jobid=157111017133827)*)
```

Bibliography

[1] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann, 1999.

[2] The Globus Project. [Online]. Available: http://www.globus.org

[3] The Globus Resource Specification Language RSL v1.0. [Online]. Available: http://www-fp.globus.org/gram/rsl%5Fspec1.html

[4] The NorduGrid Project. [Online]. Available: http://www.nordugrid.org

[5] A. Wäänanen, "An Overview of an Architecture Proposal for a High Energy Physics Grid," in *Proc. of PARA 2002, LNCS 2367, p. 76*, J. Fagerholm, Ed. Springer-Verlag Berlin Heidelberg, 2002.

[6] A. Konstantinov, "The NorduGrid project: Using Globus toolkit for building Grid infrastructure," *Nucl. Instr. and Methods **A** 502, pp. 407-410*, 2003.

[7] M. Ellert. The NorduGrid toolkit user interface. [Online]. Available: http://www.nordugrid.org/documents/NorduGrid-UI.pdf

[8] A. Konstantinov. The NorduGrid Grid Manager. [Online]. Available: http://www.nordugrid.org/documents/GM.pdf

[9] Open source implementation of the Lightweight Directory Access Protocol. [Online]. Available: http://www.openldap.org

[10] Globus replica catalog service. [Online]. Available: http://www-fp.globus.org/datagrid/replica-catalog.html

[11] Replica Location Service. [Online]. Available: http://grid-data-management.web.cern.ch/grid-data-management/replica-location-service/index.html

Index