

# Job submission on a Hemeris cluster

## User's Manual

### Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Administrative procedures</b>	<b>3</b>
<b>3</b>	<b>Technical Procedures</b>	<b>3</b>
3.1	The environment . . . . .	3
3.2	Binaries Availability . . . . .	4
3.3	Data Availability . . . . .	4
3.4	License Availability . . . . .	4
3.5	How to run a submission . . . . .	4
3.6	Submission attributes . . . . .	5
3.6.1	The "domain" attribute . . . . .	5
3.6.2	The "lic-product" attribute . . . . .	5
3.6.3	The "lic-file" attribute . . . . .	5
3.6.4	The "lic-host" attribute . . . . .	5
3.6.5	The "lic-log" attribute . . . . .	5
3.7	Job attributes . . . . .	5
3.7.1	The "exec" attribute . . . . .	6
3.7.2	The "type" attribute . . . . .	6
3.7.3	The "dir" . . . . .	6
3.7.4	The "params" attribute . . . . .	6
3.7.5	The "stdin" attribute . . . . .	7
3.7.6	The "stdout" attribute . . . . .	7
3.7.7	The "stderr" attribute . . . . .	7
3.7.8	The "name" attribute . . . . .	7
3.7.9	The "depends" attribute . . . . .	8
3.7.10	The "tasks" attribute . . . . .	8
3.7.11	The "email" attribute . . . . .	8
3.8	Job submission . . . . .	8
3.8.1	hsub . . . . .	8
3.8.2	hresume . . . . .	8
3.9	Job control . . . . .	9
3.9.1	hdel . . . . .	9
3.9.2	hstat . . . . .	9
3.9.3	hnotify . . . . .	9
3.10	Administrative commands . . . . .	9

3.10.1	halert . . . . .	9
<b>4</b>	<b>Generic environments</b>	<b>9</b>
4.1	The “single” environment . . . . .	10
4.2	The “array” environment . . . . .	10
4.3	The “mpich” environment . . . . .	11
4.4	The “pvm” environment . . . . .	12
<b>5</b>	<b>Specific environments</b>	<b>12</b>
<b>6</b>	<b>Annexes</b>	<b>12</b>
6.1	ssh and friends . . . . .	12
6.1.1	ssh keys . . . . .	12

## 1 Introduction

Job submission on a Hemeris cluster must follow specific procedures. There are administrative procedures and technical procedures.

Administrative procedures

- user authorisation
- user approval
- account creation
- account validation
- cluster attribution

Technical procedures

- environment
- binaries availability
- data availability
- licence availability
- job description
- job submission
- job control
- generic environments
- specific environment

Annexes

- ssh and friends

## 2 Administrative procedures

Administrative procedures are handled by your account manager.

## 3 Technical Procedures

The correct application of all technical procedures should allow submissions to run smoothly. Consequently, any technical problem is either due to a bug, a procedure error or ultimately to a faulty or unclear procedure.

### 3.1 The environment

Procedures' meanings are defined according to their environments. The most important parts of a user's environment are his home directory and his shell.

By default, your home directory's space name is unified across all clusters. By default, the home directory is the only space name you can be sure of.

Your default shell is bash, and your profile is not set. However, in specific environments, PATH and needed environment variables are set for you.

### 3.2 Binaries Availability

Binaries include not only programs, but also libraries, configuration data files and so on, i.e., anything except data related to a specific problem.

Three kinds of binaries can be found on a cluster node: system binaries, environment binaries and users binaries.

System binaries are all the binaries normally found on a “standard” system. Generally speaking, they include all traditional linux utilities and no specific programs. In the case of Hemeris clusters, this also includes parallel libraries (MPI, PVM, etc.).

Environment binaries are the binaries installed for a specific work environment setup by Hemeris team. Should you use such a “foo” environment, you can expect to have access to all foo programs and libraries, with the PATH and environment variables conveniently set up for you.

User binaries comprise all the stuff **you** have installed. You are therefore accountable for the completeness and configuration of your installation.

### 3.3 Data Availability

Except in the case of specific environments where Hemeris provides access to certain databases, data availability depends on the user. All user’s data will be available in the home directory and can be referenced from there. Data transfer can only be done over ssh. Available commands are therefore: scp, sftp and rsync -e ssh.

### 3.4 License Availability

In the case your application is unable to read a license file in your home directory, you will need to get an agreement from Hemeris to use the cluster-dedicated license server. As for now, only Hemeris staff are authorised to install a license on this server.

### 3.5 How to run a submission

To submit a job on a cluster you have to describe it. Description files are XML files. Each xml file linked to a submission is called the submission file. Submission characteristics are determined by the attributes.

Submissions also include job descriptions. There are submission attributes and job attributes. The tags <submission> and </submission> set the limits of the submission file. Within those limits, you can specify the submission’s attributes (in the submission start tag) as well as the start and stop tags of job descriptions (resp. <job and />). Job attributes must be placed between the job tags. File structure is as follows:

```
<submission
  S-attribut_0="valeur"
  ...
  S-attribut_n="valeur"
>
<job
  J-attribut_0="valeur"
  ...
  J-attribut_n="valeur"
/>
...
<job
  J-attribut_0="valeur"
  ...
```

```
    J-attribut_n="valeur"  
  />  
</submission>
```

In this file, job order is irrelevant and does not reflect execution order (but see "dependence" below). Within a given job submission, attributes order is irrelevant, though attributes should be unique in their scope (submission or job).

## 3.6 Submission attributes

### 3.6.1 The "domain" attribute

The domain attribute is used to specify the partition in which your submission will be executed. A partition represents a cluster or a subcluster, isolated from the others. Initialisation, cleaning, distributions and other parameters can be specified for each domain. You will have to make a reservation to use any domain but the default one.

Default: default

Legal values: default

Message when using illegal value: Domain not accessible

Other limits: domain reservation should be done by a Hemeris operator who will then give you the appropriate value to use.

### 3.6.2 The "lic-product" attribute

Default: none

Identifies an application licensing environment, i.e. which flexlm server and vendor daemon should be used with the submission. Usage of this attribute allows one to start a license server with his own license file. The server is started before the first job of the submission and stopped once all jobs are terminated.

### 3.6.3 The "lic-file" attribute

Default: none, attribut is mandatory if lic-product is specified.

The license file. The path is relative to the home directory, unless it starts with a '/

### 3.6.4 The "lic-host" attribute

Default: none, attribut is mandatory if lic-product is specified.

The host on wich the license server should be launched. This attribute must match the host id specified by the license file.

### 3.6.5 The "lic-log" attribute

Default: none, attribut is mandatory if lic-product is specified.

The file to which the license server output should be directed. The path is relative to the home directory, unless it starts with a '/

## 3.7 Job attributes

Here follows a description of all possible job attributes. Some of them are meaningless outside specific environments, or have their meaning altered in those environments. They will be described in the attribute description, and their exact meaning is given in the environment description.

If your submission file contains a generic XML error, the error message will give you the line and character number of the error's position and anything that the XML parser will find interesting to add. For example:  
Your submission request:5:24: not well-formed (invalid token)  
Sample submission result: submission id: followed by the submission number.

### 3.7.1 The "exec" attribute

The exec attribute defines the name of the executable you submit to the cluster. This attribute should not contain parameters related to the executable, as a separate attribute is defined to handle that.

Legal values: any executable in the PATH

(/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games by default). You may of course submit an executable from your HOME directory. Remember to prefix the executable's name with ./, as the current directory is **never** part of the PATH. If you want to use a subdir, use the dir attribute.

Default: empty

Message when using a null or undefined value: submission number and message in the error file.

Other limits: value's maximum length is 1024 characters, attribute cannot be omitted, value must have the execution bit set

### 3.7.2 The "type" attribute

The type attribute defines your job's environment. If this attribute is not defined, the default value is used.

Default: single

Generic values defined: single, array, mpich, pvm

Message when using a null or undefined value: <type>: unkown job type

Other limit: value's maximum length is 64 characters.

### 3.7.3 The "dir"

attribute

The dir attribute is the argument of a cd (change directory) which is made **before** executing your submission. It is usually a fragment of a path which starts in your HOME directory. Consequently, it is not recommended to make it an absolute path, as it will not be portable when Hemeris moves the file servers. The tilde prefix ( ) is not supported. In addition to the cd, this path fragment is pre-pended to the files sdtout and sdterr.

Default: empty (this corresponds to the HOME directory)

Legal values: a path fragment, implicitly starting in the home directory ; it must exist.

Message when using an illegal value: none, but the job fails.

Other limits: there is no shell expansion, and character set allowed is part of [-\_+a-zA-Z0-9]

### 3.7.4 The "params" attribute

The params attribute is used to define the parameters you want to pass to your executable. Some specific environments disallow certain parameters.

Default: empty

Legal values: executable dependant, sometimes environment dependant

Message when using an illegal value: executable dependant

Other limits: files have to be in the directory defined by dir or, if not in dir, be prefixed by the partial path from dir to where they are. params maximum length is 1024 characters. Please note that the string is handled by a shell, so shell expansion rules apply before execution.

### 3.7.5 The "stdin" attribute

The stdin attribute receives the name of the file from which the standard input will be redirected for job execution. This file must exist within the directory defined by the dir attribute.

Default: TBD

Message when using an illegal value: submission number, error message in the error file.

Other limits: filename must be constructed with characters from the set [-\_+a-zA-Z0-9]

### 3.7.6 The "stdout" attribute

The sdtout attribute defines the name of the file to which the standard output will be redirected for job execution. This file will be created in the directory defined by the dir attribute. If the file already exists, the new output will be appended to the file.

Default: job\_<job\_number>.out

Message when using an illegal value: submission number, error message in the job report (see the hstat command):

```
state:          failed
Reason:         RM error
```

Other limits: filename must be constructed with characters from the set [-\_+a-zA-Z0-9]. The system does not prevent several processes from trying to write simultaneously on this file. There is no lock on sdtout, so you are bound to lose data if you write simultaneously on this file.

### 3.7.7 The "stderr" attribute

The sdterr attribute defines the name of the file to which the standard error will be redirected for job execution. This file will be created in the directory defined by the dir attribute. If the file already exists, the new error will be appended to the file.

Default: job\_<job\_number>.err

Message when using an illegal value: submission number, error message in the job report (see the hstat command):

```
state:          failed
Reason:         RM error
```

Other limits: filename must be constructed with characters from the set [-\_+a-zA-Z0-9]. The system does not prevent several processes from trying to write simultaneously on this file. There is no lock on sdterr, so you are bound to lose data if you write simultaneously on this file.

### 3.7.8 The "name" attribute

The name attribute allows to name a job. This gives the job an invariant handle within the submission (the job ID is not invariant and is not known beforehand). Therefore, that handle cannot be used to define dependencies.

Default: empty

Message when using an illegal value: TBD

Other limits: maximum name length 256 characters.

### 3.7.9 The “depends” attribute

The depends attribute allows to specify by name the job(s) that have to be finished before launching the job including the depend. The referenced job(s) have to terminate cleanly with an exit code of 0 (zero).

Default: empty

Message when using an illegal value: dependencies cannot be satisfied

Other limits: dependency graph cannot be cyclic.

### 3.7.10 The “tasks” attribute

The tasks attribute is environment-dependant. In MPI or PVM jobs it represents the number of simultaneous processes requested for the job ; in an array job it corresponds to the number of executions requested. It is ignored in the single environment. As for specific environments, its exact meaning is specified in each environment’s description.

Default:

Legal values: 1-64 for parallel jobs, no limits for array jobs

Message when using an illegal value: none

Other limits: if you request an unavailable number of processors, your job will be queued until the requested number of processors are available ; this could take a very long time indeed.

### 3.7.11 The “email” attribute

The email attribute is used to specify the email address to which an end job notification should be send. The mail will contain the exit code of the job.

Default: empty

Message when using an illegal value: none, but the mail is not sent

Other limits: mail delivery depends on the good health of Hemeris mail servers, on the good health of your servers, and on the good health of the Internet parts in between. There is no guarantee of delivery. If there is a lag in delivery, the full mail header can be used to trace the origin of the problem.

## 3.8 Job submission

There are two ways of submitting a job: the general way and the resume way. In both cases, the system asks for a description of the submission and gives back a submission number. In the general way, only submission correctness is evaluated. As for the resume way, a job restart procedure is also needed, and the executable is required to support some kind of checkpointing. In practice, the procedure only exists for the environments that define it.

### 3.8.1 hsub

The hsub command is the general command used to submit a job. On the submission machine, the correct syntax is: `hsub <submission_file>` If there is no XML error in the submission file, the general result of a submission is the message: `submission id:` followed by submission number.

### 3.8.2 hresume

the hresume command is a hook to the resume command for interrupted jobs. Each environment defines how that hook can be used.

### 3.9 Job control

The control commands allows users to interact with a job and to query his state. There are two generic commands (hdel and hstat) that act at resource manager level. The other one is specific to the application being run and allows to interact with it (hnotify).

#### 3.9.1 hdel

The hdel command allows unconditional stopping of a job or a submission. Of course, the job or submission must still be running to be stopped. Syntaxes are:

```
hdel -s <submission_number>
hdel -j <job_number>
```

#### 3.9.2 hstat

The hstat command displays status for all active submissions, or for any past job/submission you own. Syntaxes are as follows:

```
hstat
hstat -s <submission_number>
hstat -j <job_number>
```

The first form gives an answer only if there is a job running.

#### 3.9.3 hnotify

The hnotify command is a hook to communicate with an application. Each environment defines how that hook can be used.

### 3.10 Administrative commands

#### 3.10.1 halert

The halert command is a last resort command which allows you to page an operator when other means to get support failed. In practice, the argument to halert is sent by SMS to the on-call operator. The normal procedure to access support is and remains either mail (support@hemeris.com) or phone +322 289 0002).

```
halert <message to send>
```

Where <message to send> is a string of characters which is sanitised for SMS consumption: the string is converted in iso-latin15 and truncated to 160 characters. Remember to include your contact co-ordinates at the start of the message. Do not try to send 2 SMS to explain what your problem is, do it by email and/or wait for the operator to contact you.

## 4 Generic environments

Generic environments provide all you need to run your applications on a cluster. As those environments are not linked to any vendor, we can provide them on a permanent basis. In some cases, use of a generic environment may prove to be "too generic" and you might have to install too many binaries in your home directory. Should the environment you create in this process be of interest to others, you can suggest that we make it a generic one. You would then be freed of environment maintenance, but you would also lose some control over it.

#### 4.1 The “single” environment

It is the simplest of all environments. It is basically the submission of a single executable that will be run on one node. Say for example that you have the following file `test.sh`:

```
#!/bin/sh
echo "Hello from 'hostname'"
```

And, given the following submission file `sub.xml`:

```
<submission>
  <job
    exec="./test.sh"
    stderr="hello.err"
    stdout="hello.out"/>
</submission>
```

Session results would be:

```
njtest@willkommen:~$ hsub sub.xml
submission id: 35
njtest@willkommen:~$ cat hello.err
njtest@willkommen:~$ cat hello.out
Hello from n008
```

Please note that neither the “submission id” value nor the node’s name are significant. On the contrary, it is mandatory that `test.sh` should be prefixed like `./test.sh`. Remember that the current directory is not in the `PATH`, and that you have to reference it explicitly. Your default `PATH` is `/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games`, as you can see by submitting a single job with the command `echo "$PATH"`. You can modify your `PATH` if desired ; both `.bashrc` and `.bash_profile` are available.

#### 4.2 The “array” environment

The array environment is most suited to embarrassingly parallel tasks. The `task` attribute is then used to specify the number of times the job should be launched. To allow the job to generate a different result each time, the environment variable `TASK_ID` is defined, initialised at “1” and incremented at each iteration. Say that we have the following files, `test_a.sh` and `test_a2.sh`:

```
#!/bin/sh
echo "Hello from 'hostname'" > result_${TASK_ID}.txt
```

```
#!/bin/sh
if [ -e result_tot.txt ]; then rm result_tot.txt; fi
for i in result_*.txt
do
  cat $i >> result_tot.txt
done
```

And, given the following submission file `sub_a.xml`:

```

<submission>
  <job
    name="do_all"
    exec="./test_a.sh"
    type="array"
    tasks="10"
    stderr="hello_a.err"
    stdout="hello_a.out"/>
  <job
    depends="do_all"
    exec="./test_a2.sh"/>
</submission>

```

Session results would be:

```

njttest@willkommen:~$ hsub sub_a.xml
submission id: 47
njttest@willkommen:~$ ls result*
result_1.txt  result_2.txt  result_4.txt  result_6.txt  result_8.txt  result_tot.txt
result_10.txt result_3.txt  result_5.txt  result_7.txt  result_9.txt
njttest@willkommen:~$ cat result_tot.txt
Hello from n014
Hello from n025
Hello from n015
Hello from n016
Hello from n009
Hello from n020
Hello from n021
Hello from n022
Hello from n024
Hello from n018

```

This example also demonstrates the usefulness of the “name” and “depends” attributes

### 4.3 The “mpich” environment

The installed version of MPI is mpich-1.2.5, which is in /usr/local/mpich-1.2.5. Let us say that you want to submit the cpi example of the distribution, and the following submission file sub\_m.xml:

```

<submission>
  <job
    exec="./cpi"
    type="mpich"
    tasks="8"
    stdout="cpi.out"
    stderr="cpi.err"/>
</submission>

```

Session results would be:

```
njtest@willkommen:~$ hsub sub_m.xml
submission id: 48
njtest@willkommen:~$ cat cpi.out
/usr/local/sge/default/spool/n027/active_jobs/44.1/pe_hostfile
n027
n019
n017
n028
n029
n005
n003
n006
pi is approximately 3.1416009869231245, Error is 0.0000083333333314
wall clock time = 0.004051
```

#### **4.4 The “pvm” environment**

The installed version of PVM is PVM 3.4.4, which is under `/usr/local/pvm3`.

### **5 Specific environments**

Depending on your contract with Hemeris, you might have access to some specific environments. They are adapted by Hemeris to the needs of a specific program or program suite. Your submission’s parameters may have different meanings in such an environment. All these modifications are referenced in the appropriate description.

### **6 Annexes**

#### **6.1 ssh and friends**

The purpose of this annex is not to duplicate the documentation of ssh, but to give you a short “getting started”.

##### **6.1.1 ssh keys**

ssh uses a public key/private key system. You generate a pair of keys with the command `ssh-keygen`, specifying the desired kind of key ; you get two files with the same names, except that one ends with `.pub`. The file ending with `.pub` is your public key, the other one is your private key. When generating the keys, you have the opportunity to protect the private key with a passphrase. The confidentiality of your private key depends on two factors: the degree of confidentiality of the physical support where it is stored and the length and complexity of your passphrase. As soon as you are confident with the way you protect your private key, all you need to do is to append your public key in the file `authorized_keys` in the `.ssh` directory of any host to which you want to access with your (private) key, i.e., without password.

If you did what is described in the first paragraph, you swapped "identification by password" with "identification by key". Each time you use your key, you will have to issue your passphrase instead of your password, which is not always convenient. Therefore, you have the option to use a ssh agent. This is a program that keeps your keys unlocked in memory for a defined amount of time, but only as long as your local session is alive.

Say that you are in a local bash session ; the following commands will generate the keys, install the (public) key in our access point and register your private key with your local agent.

```

nj@nicolas:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/nj/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/nj/.ssh/id_rsa.
Your public key has been saved in /home/nj/.ssh/id_rsa.pub.
The key fingerprint is:
74:ea:5e:1d:2a:3c:ad:b3:51:12:3c:db:56:f8:43:2a nj@nicolas

nj@nicolas:~$ scp /home/nj/.ssh/id_rsa.pub
njtest@willkommen.hemeris.com:~/.ssh/authorized_keys
The authenticity of host 'willkommen.hemeris.com (212.233.8.52)'
cannot be established.
RSA key fingerprint is 01:33:83:96:a9:44:54:2c:59:1b:f5:b2:04:9a:5b:c8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'willkommen.hemeris.com' (RSA) to the list of known hosts.
njtest@willkommen.hemeris.com's password:
id_rsa.pub          100% 220   759.1KB/s   00:00
nj@nicolas:~$ ssh-agent bash
nj@nicolas:~$ ssh-add
Enter passphrase for /home/nj/.ssh/id_rsa:
Identity added: /home/nj/.ssh/id_rsa (/home/nj/.ssh/id_rsa)
nj@nicolas:~$ ssh -l njtest willkommen.hemeris.com
Last login: Tue Nov  4 14:03:38 2003 from 212.68.250.204.brutele.be
njtest@willkommen:~$ cd .ssh/
njtest@willkommen:~/.ssh$ ls
authorized_keys  id_dsa  id_dsa.pub  id_rsa  id_rsa.pub  known_hosts

```

**Important:** the other keys already installed in your .ssh directory on willkommen are the keys used to access the submission engine, do not erase them.