LHC COMPUTING GRID

EXPERIMENT SOFTWARE INSTALLATION IN LCG-2

MANUALS SERIES

Document identifier:	**Non applicable**
EDMS id:	**None**
Version:	**1.0**
Date:	**July 22, 2005**
Section:	**LCG Experiment Integration and Support**
Document status:	**PUBLIC**
Author(s):	Roberto Santinelli, Simone Campana
File:	**sw-install**

Abstract: *About the installation of experiments software on LCG-2 sites*

Document Change Record

Issue	Item	Reason for Change
22/07/05	v1.0	First version

Files

Software Products	User files
PDF	http://grid-deployment.web.cern.ch/grid-deployment/eis/docs/ExpSwInstall/sw-install.pdf
HTML	http://grid-deployment.web.cern.ch/grid-deployment/eis/docs/ExpSwInstall/sw-install.html

CONTENTS

1. INTRODUCTION

1.1. ACKNOWLEDGEMENTS

This work received support from the following institutions:

- Istituto Nazionale di Fisica Nucleare, Roma, Italy.
- Ministerio de Educacion y Ciencia, Madrid, Spain.

REFERENCES

[R1] Experiment Software Installation
https://edms.cern.ch/file/498080/1.0/SoftwareInstallation.pdf

[R2] Tank And Spark
http://grid-deployment.web.cern.ch/grid-deployment/eis/docs/internal/chep04/SW_Installation.pdf

[R3] How to Manually configure and test the Experiment Software Installation mechanism on LCG-2
http://grid-deployment.web.cern.ch/grid-deployment/eis/docs/configuration_of_tankspark

[R3] LCG-2 User Guide
https://edms.cern.ch/file/454439//LCG-2-UserGuide.html

1.2. OVERVIEW

Authorized users can install software in the computing resources of LCG-2. The installed software, which we will call ***Experiment Sofware***, is also published in the Information Service, so that user jobs can run on nodes where the software they need is installed.

In [R1], a collection of official requirements for the procedure to install software on LCG-2 nodes from both the experiments and the site administrators is given. In the same document, a possible approach to the problem is described. This approach represents the starting point of the mechanism described in this document.

2. GENERAL PROCEDURE

The Experiment Software Manager (ESM) is the member of the experiment VO entitled to install Application Software in the different sites. The ESM can manage (install, validate, remove...) Experiment Software on a site at any time through a normal Grid job, without previous communication to the site administrators. Such job has in general no scheduling priorities and will be treated as any other job of the same VO in the same queue. There would be therefore a delay in the operation if the queue is busy.

The site provides a dedicated space where each supported VO can install or remove software. The amount of available space must be negotiated between the VO and the site.

An environmental variable holds the path for the location of a such space. Its format is the following:

`VO_<name_of_VO>_SW_DIR`

Different site configurations yield different scenarios for software management:

- A file system is shared among all WNs of the same cluster: the software is installed in the file system, and the jobs that want to use it just need to use the `VO_<name_of_VO>_SW_DIR` variable to access the software through POSIX calls.
- The site does not provide a shared file system; the environment variable `$VO_<name_of_VO>_SW_DIR` contains a ".". There are in this case two different possibilities depending on whether the site provides the Tank & Spark ([R2]) mechanism for automatic software propagation or not:
  1. **WITH T & S**: the software gets permanently installed on each WN.
  2. **WITHOUT T & S**: Bundles of software are stored in the closest SE in the site, and they get only installed in a WN when a job requires them. After the job finishes, the software is automatically erased by the batch system from the WN. This is done to avoid the exhaustion of disk space in the node.

Once the software is installed, the ESM can perform the validation.

The validation is meant as a process or a series of processes and procedures that verify(ies) the installation of the software. It can be performed in the same job, after installation (validation *on the fly*) or later on, via a dedicated job.

If the ESM judges that the software is correctly installed, it publishes in the Information System (IS) the tag which identifies unequivocally the software. Such tag is added to the `GlueHostApplicationSoftwareRunTimeEnvironment` attribute of the IS, using the GRIS running in the CE. Jobs requesting for a particular piece of software can be directed to the appropriate CE just by setting special requirements on the JDL.

3. IMPLEMENTATION DETAILS

LCG has implemented the schema described above with a three layers software as summarized in Figure 1:

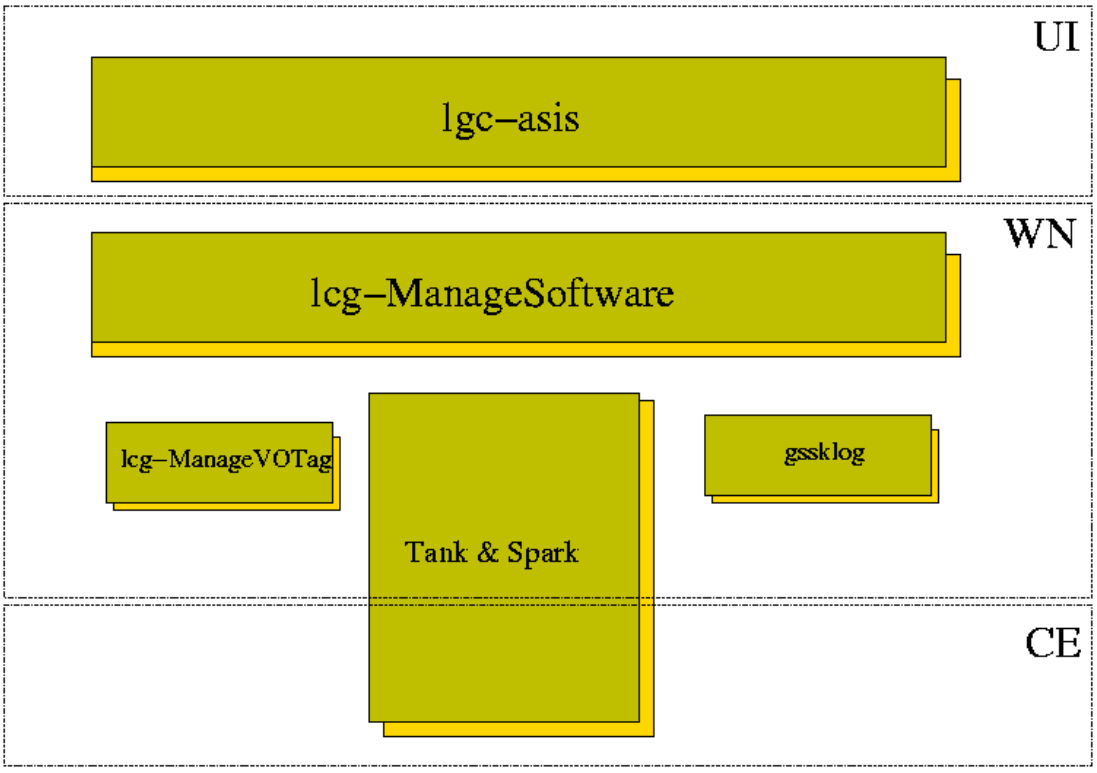

Figure 1: Multi layer structure of the current implementation of the Experiment Software Installation toolkit

The top layer, `lcg-asis`, is the friendly interface that can be used by the ESM to install/validate or remove software on one or more sites that are allowed for the VO the ESM belongs to.

The middle layer, installed on each WN, `lcg-ManageSoftware` is the toolkit that steers any Application Software management process.

Finally, the low level layer is constituted by `lcg-ManageVOTag`, which is used to publish tags on the Information System, `Tank&Spark`, which is used to propagate software in case the site doesn't provide shared disk space and the `gssklog` client, which is used to convert GSI credential into AFS Krb5 tokens.

The ESM must prepare a tarball(s) containing the software to be installed and some scripts to install, validate, and remove the software. No constraints are imposed to the names of these scripts and tarballs. Afterwards, the ESM invokes (from the UI) the `lcg-asis` script by providing the name of the tarballs

and/or the name of the scripts to be used for installing, removing or validating the software. If no tarballs are specified, the scripts will install software directly from some central repositories[1].

The `lcg-asis` script performs basically the following actions:

- Uploads the specified tarball (if any) on the grid (`lcg-cr` command).
- Loops over suitable sites for the given VO matching the requirements of the ESM.
- For each site checks whether other installation (or validation or removal) processes are running on the site.
- For each site composes the JDL with the adequate parameters (see later for more details).
- For each site submits the job.

These special jobs submitted either with `lcg-asis` or manually, invoke on the target WN the script `lcg-ManageSoftware`, which performs the following actions:

- Checks whether other processes are running on the site by seeking on the Information System the existence of special temporary tags. (see later)
- Publishes a temporary tag on the Information System informing that there is a management process running on the site (using `lcg-ManageVOTag`)
- Sets up a special temporary installation area that will be cleaned up at the end of the job and that is used to store temporary files used in the installation/validation/removal process.
- Checks whether the site supports a shared file system (including AFS) or the site doesn't support a shared file system but has installed Tank & Spark.
- Checks for the existence on the grid of the specified tarballs.
- Replicates these tarballs to the closest Storage Elements and copies from there to the WN (no outbound connectivity is required in this way).
- Runs the tool to convert the GSI credentials into valid AFS Kerberos tokens whenever there is an AFS area.
- Unpacks these tarballs into a temporary directory if no scripts has been shipped with the Input Sandbox. (Supported formats: `*.tgz`, `*.tar.gz`, `*.tar`).
- Calls the appropriate script as specified by the user and checks the return code of the script itself.
- Cleans up the temporary files used in the installation/removal/validation process.

[1] In this case, the ESM assumes the *outbound* connectivity from the site

- Calls the client of the Tank & Spark service (if running) in order to trigger the propagation of the software to the rest of the components of the farm.
- Publish another temporary flag informing about the progress (or the failure) of the process itself.

4. FLAVOURS OF THE PUBLISHED TAGS

The progress of each Experiment Software management process, is updated and/or monitored by each one of the software layers provided by LCG for the Experiment Software Installation. It relies upon a schema of *flavours* for the TAG that is published on the Information System, for any given version of Experiment Software that requires to be installed/removed/validated. The format of the flag which is going to be published at every change of the status of the undergoing process is:

```
VO-<name_of_VO>-<flag-provided-by-ESM>-<status-flavour>
```

The *flavour* is a string that informs the ESM about the status of a given Experiment Software management process. It can be one of these possible values:

- `processing-install`: the software identified by `<flag-provided-by-ESM>` is getting installed.
- `processing-remove`: the removal process of the software is running.
- `processing-validate`: the validation process of the software is running.
- `aborted-install`: the installation process got aborted.
- `aborted-remove`: the removal process got aborted.
- `aborted-validate`: the validation process got aborted.
- `to-be-validated`: the installation process succeeds to run and the flag is now ready to be validated.

There are still two other possible states:

- Standalone TAG (without added flavors): the validation process runs successfully on the site and the software can be used by all the users.
- TAG no longer published : the removal process runs successfully and the software has been fully removed from the site.

Using such flag publication mechanism is useful to avoid concurrent processes running simultaneously for the same Application Software.

5. LCG-ASIS

The tool exposes some features that help the ESM in his activity.[2] It allows the VO software Manager to gather information about the software installed in all sites supporting the VO (*Querying* operational mode). Besides, it can automatically build the JDL files used for managing experiment software on a site, store them in a directory (*Managing* operational mode) and even send the corresponding jobs. It will be duty of the Software Manager to verify they ended correctly and collect the output.

The former operational mode is invoked with the command (from a UI) `lcg-asis --status`. This mode requires a set of *general-options* and some other specific *status-options*.

One can run `lcg-asis` in a *Managing* operational mode using the options `--install`, `--validate` or `--remove` and passing some other specific (*manage-options*) together with the general options. The `--install` and `--validate` options can be used simultaneously to install software and validate it on the fly.

general-options

- `--tag TAG`: Specifies the tag which identifies the package. If used with the `--status` switch, accepts also `all` as argument, showing the status of all tags for the given VO.
- `--vo VO`: Specifies the name of the VO the ESM belongs to.
- `--os OS`: Specifies the Operative System type for which the package is supported. Possible values are `RH7` (RedHat7), `SL` (Scientific Linux), `other` (none of the previous). Mandatory for `--install`, `--validate`, `--remove`; auxiliary for `--status`.
- `--bdii HOSTNAME:PORT`. By default the BDII specified by the ENV variable `LCG_BDII_INFOSYS` is used. This option allows to specify a different one.
- `--filter EXPR` Allows to reduce the list of clusters to the ones matching the expression `EXPR`. The filtering is performed through the `lcg-info` tool (use `lcg-info --help` for details in respect of filtering attributes and syntax).

The `--filter` option requires a bit more exhaustive explanation. The syntax of this option inherits from the `lcg-info` command. The list of attributes which can be used to make a filter are listed by issuing the command:

```
$ lcg-info --list-attrs
```

An example of a filter that looks for CEs that have at least one CPU free and no more than one waiting jobs follows:

[2] The tool has not yet been deployed with the rest of the toolkit; it is available for download at the link: http://grid-deployment.web.cern.ch/grid-deployment/eis/docs/lcg-asis

`--filter "FreeCPUs >=1,WaitingJobs <=1"}`

The comma is interpreted as an *AND*. No blank spaces must be included in the expression. Possible operators are: `=, >=, <=`

status-options (used with the `--status` options)

- `--verbose`: Increases verbosity of the output. If you specify a tag and you pass `--verbose` option your output will show all tags. The specified tag will be marked as selected. Furthermore, the query will return all possible CEs and not just the ones selected by your filter.

manage-options (used with the `--install`, `--validate` and `--remove` options)

- `--install-script NAME`: Specifies the installation script. Such script will be available at runtime in the Worker Node and must drive the installation process
- `--validate-script NAME`: Specifies the validation script. Such script will be available at runtime in the Worker Node and must drive the installation process
- `--remove-script NAME`: Specifies the removal script. Such script will be available at runtime in the Worker Node and must drive the installation process
- `--upload FILENAME`: Upload a file to the grid. This file will be available at runtime in the Worker Node. In order to upload more than one file, the option can be used multiple times.
- `--time TIME`: By default the job is required to run in the queue with longer `GlueMaxWallClockTime`. The `--time` option drives the job to the queue with the smaller `GlueCEMaxWallClockTime` but greater than `TIME` (in minutes). Useful if the ESM knows how long the Installation process takes.
- `--basedir BASEDIR`: JDL jobs are created under the directory `basedir/dirname`. By default `basedir` is the current directory. The `--basedir` option changes the default to the `BASEDIR` value.
- `--dirname DIRNAME`: JDL jobs are created under the directory `basedir/dirname`. By default `dirname` has the form `lcg-asis_<action>_<date>_<time>_<vo>_<TAG>_<OS>`.
- `--notify EMAIL-ADDRESS`: For each site with Tank & Spark mechanism in place, a notification email is sent to `EMAIL-ADDRESS`, with the details of the operation performed (both in case of success or failure).

6. LCG-MANAGESOFTWARE

The command `lcg-ManageSofwtare` must be invoked by providing several arguments that must be passed in the `Arguments` field of the JDL.

First, the action to perform must be specified. This will take one of the following values:

1. `--install`: for installing a new software on the CE.
2. `--uninstall`: for removing a software on the CE.
3. `--validate`: for validating a software already installed on the CE.
4. `--iv` or `--install --validate`: for installing and validating with the same job a new software on the CE.
5. `--run` for running a given Application Software.

There are quite few other informations that should be provided.

- `--vo <user-vo>`: is the VO of the ESM that runs the job.
- `--tag <flag-provided-by-the-experiment>`: is a string that unequivocally identifies the software. The header (`VO-<vo-name>`) and footer (eventual flavours) will be automatically appended by the script. It is not mandatory if both the `-S` and `-V` options are specified.
- `--tar <tarball(s)>`: is the tarball (or the list of tarballs) to be downloaded from the GRID. If there are more than one tarball, the list of tarballs must be enclosed between single quotes. It is not mandatory if the installation (or validation or removal) script does not require it. In this case the script must be shipped with the input sandbox.
- `--notify <your-email-address>`: is the ESM e-mail address used eventually by Tank to notify about the result of the undergoing process.
- `--install_script <name-of-the-script>`: is the name of the installation script that will be invoked by `lcg-ManageSoftware`. If not specified, the default value `install_sw` is taken into account.
- `--uninstall_script <name-of-the-script>`: is the name of the removal script that will be invoked by `lcg-ManageSoftware`. If not specified, the default value `uninstall_sw` is taken into account.
- `--validate_script <name-of-the-script>`: is the name of the validation script that will be invoked by `lcg-ManageSoftware`. If not specified, the default value `validate_sw` is taken into account.

- `--run_script <name-of-the-script>`: is the name of the run script that will be invoked by `lcg-ManageSoftware` to run the application. If not specified, the default value `run_sw` is taken into account.
- `--SE <name-of-the-SE>`: is the Storage Element from where the user job fetches the tarballs containing the Experiment Software. If not provided, the tool will use the default Storage Element from the environment variable `$VO_name_of_vo_DEFAULT_SE`; and if this variable is not defined it will use `edg-brokerinfo` to retrieve the information.
- `--CE <name-of-the-CE>`: is the name of the Computing Element where the Software Manager will install the software. This option becomes mandatory if the job is submitted with `-r` option due to the fact that the tool retrieve informations of the CE through the BrokerInfo.
- `-S <software-name>` (used in the past versions): is a string that informs about the name of the software to be managed.
- `-V <version>`: (used in the past versions): is a string that identifies the major version number of the software.
- `-R <release>`: (used in the past versions): is a string that identifies the minor version number (patch number) of the software. Default is `NULL`.

7. LCG-MANAGEVOTAG

The `lcg-ManageVOTag` command can be used to manage the contents of the `GlueHostApplicationSoftwareRunTimeEnvironment` attribute of the IS, for a particular CE.

Its different options can be used to add (`--add` option) a new tag for installed software, show the list of published tags for a given VO (`--list` option), remove (`--remove` option) a defined tag, or clean (`--clean` option) everything that had been previously published.

This tool can be handy in several situations, such as when an error in the information published has to be corrected, or when the tag for some software needs to be manually added. Notice that this will not be the case if the `lcg-ManageSoftware` command was used, since that tool automatically publishes the information tag for an installed and validated software. However, if the ESM chooses to use another method for the installation (e.g.: asking the site administrator to manually install the software), then he must make sure that the software is correctly published in the IS; and for that he can use the `lcg-ManageVOTag` command.

The following arguments have to be passed:

- `<-vo>` is the Virtual Organization the user belongs
- `<-host>` is the Computing Element where publish the flag
- `<-tag>` is the tag to be published/removed. The only constraint is in the format of the tag: it must always have a header `VO-<vo_name>`.

The structure of the TAG indicating a software available on a given site looks like:

```
VO-dteam-lcg_utils-4.5
```

8. GSSKLOG

If a site provides Experiment Software Area through AFS, the job owned by the ESM must be able to write on such area. Access to AFS can be authorized via an AFS token or by the client machine IP. By default, jobs that come in through the Globus gatekeeper, do not have an AFS token, and even if they did, most batch systems do not propagate tokens from the head node to the worker node. `gssklog` takes a grid proxy, authenticates with a `gssklogd` server, and obtains AFS tokens. This operation is fully hidden to the end-user by `lcg-ManageSoftware`. In order to use it, the Spark configuration file is required to be correctly set up on the WN.

9. TANK & SPARK

For the sake of completeness, the ***Tank & Spark*** mechanism is shortly described in this section. More detailed information about its design and functionality and about how to install and configure it are available in [R2] and [R3].

Tank & Spark is a mechanism based on the P2P technology that allows for a synchronization of the Software among WNs on sites that do not provided dedicated space through a shared area.

Spark is the client, called by `lcg-ManageSoftware` and run in a WN. It contacts a server (***Tank***) listening on the CE, in order to register the information about a new tag to be propagated (removed) into the server database. It also receives from Tank the coordinates of the repository of the Experiment Software (usually located in a SE). Spark synchronizes the local directory with the mirrored one in the repository, using `r-sync`.

On the other hand, the Tank service receives periodic requests for latest version of software from every WN of the site and for every VO. These are triggered by cronjobs installed on the WN. After the registration made by Spark, this requests trigger the synchronization from the central repository to the local areas of all the WNs. The software is installed in all the WNs.

Any application that needs to integrates with this service can invoke the client with the following syntax:

```
<LCG_LOCATION>/sbin/lcg-spark -h https://<HOST_CE>:18085 -a <action> -f <TAG> \
--vo <voname> -n <email_address>
```

where:

- `-h SERVICE`: is the secure service to be contacted with the format `https://<name_host>:PORT`.
- `-a ACTION`: is the action to be done. Possible values: `addflag`, `removeflag`, `upgradehost`, `updatehost`, `removehost`, `registerhost`, `get_afs_token`.
- `-f TAG`: is he tag identifying the software to be propagated/removed.
- `--vo VONAME`: is the name of the vo the ESM belongs to.
- `-n EMAIL_ADDRESS`: is the address to which any notification will be sent to.

10. EXAMPLES

This section describes a simple sample script for installing a new software on LCG-2. Guidelines about how to build (form the experiment side) the necessary scripts are also provided. A full complete list of examples that make use of the current implementation of the Experiment Software Installation mechanism can be found in [R3].

10.1. SCRIPT FROM THE EXPERIMENTS

How should the script from the experiment be created? There are only few recomendations to be taken into account while writing the script:

- the software has to be relocatable. This means that everything must be written under the directory `$VO_<name_of_VO>_SW_DIR`. The framework for managing software retrieves automatically this variable for the interested experiment. However, if this variable is a "." and if the site has installed Tank & Spark, `lcg-ManageSoftware` gets the value of the root directory from the configuration file used by spark (`/opt/lcg/etc/spark.conf`) by looking at the content of the field `<vo_name>_locdir`.

  NOTE: In order to make it fully transparent to the end user, `$VO_<name_of_VO>_SW_DIR` represents the root directory under which to install software and should be considered by the script.

- The script must return `0` in case of success of the process, or something else in case of failure in order to let the calling framework know about the result of the process and let it adopt the adequate procedure. The ESM knows which instruction that can decretate a failure or a success for a given process (installation, validation, removal) is.

- the framework, takes care also for a reliable copy of the tarballs (to be used by the script just in case) into a temporary directory of the WN. If the framework does not find the script (specified by the option `--install_script` or just the `install_sw` default) shipped with the Input Sandbox, it tries to unpack these tarballs.

  At the end of this preparation, in this temporary directory there are one (or more) tarballs (packaged or unpackaged) and the script for installing (or validating or removing) software.

An example of a such script (`install_lcg`) follows:

```
#!/bin/bash
export TAR_LOC=`pwd`   # this is the temporary directory where it is
                       # supposedd to be the steering script and the tarballs

wget http://grid-deployment.web.cern.ch/grid-deployment/eis/docs/lcg_util-client.tar.gz
                       # In this case the script doesn't need tarball from the grid but
                       # it fetches from the WEB  requiring OUTBOUND connectivity
```

```
cd $VO_DTEAM_SW_DIR    #software installation root directory
mkdir lcg_utils-4.5
cd  lcg_utils-4.5
echo "running the command : tar xzvf $TAR_LOC/lcg-util-client.tar.gz"
tar xzvf $TAR_LOC/lcg-util-client.tar.gz
if [ ! $? = 0 ]; then #failure?
 exit $?
endif
echo "running the command : tar xzvf $TAR_LOC/lcg-util-client-install.tar.gz"
  # lcg-util-client-install.tar.gz is another tarball that has been uploaded on
  # the grid and that the framework for installing software has automatically
  # replicated  on the $TAR_LOC directory of the WN.

tar xzvf $TAR_LOC/lcg-util-client-install.tar.gz
exit $?        # This is the relevant return code
```

10.2. USING LCG-MANAGESOFWTARE DIRECTLY

Once the script used for steering the installation has been made, the ESM has to create special JDL and submit it. This operation must be reiterated for all sites supporting his VO and that have not yet other process running there for the specific software. A typical JDL looks like:

```
Executable = "/opt/lcg/bin/lcg-ManageSoftware";
InputSandbox = {"install_lcg"};
OutputSandbox = {"stdout", "stderror"};
stdoutput = "stdout";
stderror = "stderror";
Arguments = "--install --install_script install_lcg --vo dteam /
  --tag lcg_utils-4.5 --tar lcg-util-client-install.tar.gz --notify support-eis@cern.ch";
Requirements = other.GlueCEUniqueID == "lcb0706.cern.ch:2119/jobmanager-pbs-long";
```

This JDL says that `lcg-ManageSoftware` is called on the WN of the site `lxb0705.cern.ch` for installing a new software (`--install` action) within the VO `dteam`. The new software is identified by the tag `lcg_utils-4.5`. From the option `--install_script`, we see that the script from the experiment that must drive the installation is `install_lcg` and at the end of the process, in case Tank & Spark is running on the site, a report of the process has to be sent to the `support-eis@cern.ch` mail address.

10.3. USING LCG-ASIS

The use of `lcg-asis` hides the difficulties that using directly `lcg-ManageSoftware` could imply. The steps summarized in the previous section are fulfilled in one go with the command:

```
lcg-asis --install --tag lcg_utils-4.5 --vo dteam --install-script install_lcg /
              --tar lcg-util-client-install.tar.gz
```

The command will visualize a template of the JDL and will let the ESM decide whether proceed, modify some field or abandon the operation.

```
Creating JDL files for job installation....

-------------------------------------------------------------------------------

Executable="/bin/bash";
OutputSandbox={"stdout","stderr"};
InputSandbox={"/afs/cern.ch/user/s/santinel/scratch0/grid/SInstallation/lcg-utils/install
 _lcg"};
stderror="stderr";
stdoutput="stdout";
Arguments="$LCG_LOCATION/bin/lcg-ManageSoftware --vo dteam --tag lcg_utils-4.5 /
           --notify roberto.santinelli@cern.ch --install --install_script install_lcg /
           --tar lcg-util-client-install.tar.gz";
Requirements = other.GlueCEUniqueID == "<CENAME>";
-------------------------------------------------------------------------------

(P)roceed with JDL creation,(M)odify attr. value, (E)xit program: P
```

In case the ESM agrees with this JDL, the tool will loop over the selected sites complying with the specified requirements (`--os`, `--time` and `--filter` options) and for each site, will create the JDL (`<CENAME>.jdl`) into the specified subdirectory. If another TAG (in whatever flavour) is already found on the site, the tool doesn't create the JDL for this specific site.

```
JDL files will be created under the directory:
/afs/cern.ch/user/s/santinel/scratch0/grid/SInstallation/lcg-utils/lcg-asis_install
_20-04-05_13:59:00_dteam_lcg_utils-4.5


*******************************************************************************
Cluster                                 OS          OSRel.      Deduced OS
lxb0706.cern.ch                         Redhat      7.3         RH7

Selected Queue: lxb0706.cern.ch:2119/jobmanager-pbs-infinite
Matching TAG(s): NO-TAG

OK: preparing JDL....
*******************************************************************************
```

Once these JDLs have been created, the ESM is prompted to decide whether to submit these jobs or leave it for later on (with `edg-job-submit` or the `submitter_general` tool see Section [R3]).

```
*********************************************************************************
The JDL files have been created under the directory
/afs/cern.ch/user/s/santinel/scratch0/grid/SInstallation/lcg-utils/lcg-asis_install_ /
20-04-05_16:30:15_dteam_lcg_utils-4.5

Do you wish to submit the jobs? [Y/n]Y
*********************************************************************************
```

If the ESM decides to let the tool submit the jobs (Y), the jobs get submitted and all Job-IDs will be stored in to a file under the `basedir/dirname` directory called `submitted-job-id.list`.

Tip: Instead of installing software around all sites the ESM might want to install just in somes sites, whose cluster name he knows. In this case the options to pass are:

```
lcg-asis --vo dteam -tag lcg_utils-4.5 --status --verbose \
-filter "Cluster=lxb0706.cern.ch"  --time 150
```

In this example, the ESM will just install on the site `lxb0706.cern.ch` and the job will be submitted to the queue `lxb0706.cern.ch:2119/jobmanager-pbs-long` whose the `MaximumWallClockTime` is the shortest but it's also greater that 100 minutes (prioritizing issue).

```
*********************************************************************************
Cluster              OS           OSRel.       Ded.OS       Q selected?     Tag status
lxb0706.cern.ch       Redhat       7.3          RH7          YES             NO-TAG

Queue                                                          MaxWCT
lxb0706.cern.ch:2119/jobmanager-pbs-short                      5
lxb0706.cern.ch:2119/jobmanager-pbs-infinite                   2880
lxb0706.cern.ch:2119/jobmanager-pbs-long      (Selected)       180
```

In the last output it is possible to note that the status for the TAG `lcg_utils-4.5` is NO-TAG indicating that on the CE selected there is not the tag published in any flavour.

After the installation of the software the same command returns:

```
*********************************************************************************
Cluster              OS           OSRel.       Ded.OS       Q selected?     Tag status
lxb0706.cern.ch       Redhat       7.3          RH7          YES             SINGLE-TAG
Queue                                                                          MaxWCT
```

```
lxb0706.cern.ch:2119/jobmanager-pbs-short                                    5
lxb0706.cern.ch:2119/jobmanager-pbs-infinite                                 2880
lxb0706.cern.ch:2119/jobmanager-pbs-long          (Selected)                 180

            Tags
            VO-dteam-oleole
            VO-dteam-stat-accep-test-g4.7.0
            VO-dteam-lcg-utils-1.2.2-aborted-install
            VO-dteam-lcg-utils-2.1.5
            VO-dteam-lcg-utils-3.0
            VO-dteam-lcg-utils-3.2
            VO-dteam-lcg-utils-2.3.0-aborted-remove
            VO-dteam-lcg-utils-2.1.6-processing-remove
            VO-dteam-lcg-utils-4.1
            VO-dteam-lcg_utils-4.5-to-be-validated                       (Selected)
*******************************************************************************
```

Any attempt of run `lcg-asis` with `--install` option will fail because there is already a tag on the CE (`SINGLE-TAG` status).