Enabling Grids for E-science in Europe

www.eu-egee.org

First Latin-american Grid Workshop
Mérida, Venezuela, 15-20 November 2004

Software Installation Services

Patricia Méndez Lorenzo
patricia.mendez@cern.ch
LCG Experiment Integration and Support
CERN IT-GD/EIS

EGEE is a project funded by the European Union under contract IST-2003-508833

How I can run my private application in a world wide Grid?

How can I be sure that the software I need is installed at a remote resource where I might not have an account?

Contents

- The problem
- Requirements from the site administrators
- Requirements from the experiments
- The current mechanism
- Lcg-ManageSoftware and

  lcg-ManageVOTag

- Hands on

The problem

- The middleware software is often installed and configured by system administrators at sites via customized tools like ***LCFGng*** or ***Quattor*** that provide also for centralized management of the entire computing facility.

while

- Gigabytes of Virtual Organization (VO) specific software need too to be installed and managed at each site. As of today a flexible tool to reliably do so is not available.

The problem

- The difficulty here is mainly connected to the existence of several ways of installing, configuring and validating experiment software.
- Each VO (LHC Experiment and not only) wants to maintain its own proprietary method for distributing the software on Grid.
- Furthermore the system for accomplishing this task can change with time even inside the same VO.
- Each single user could require to have his own software installed everywhere.

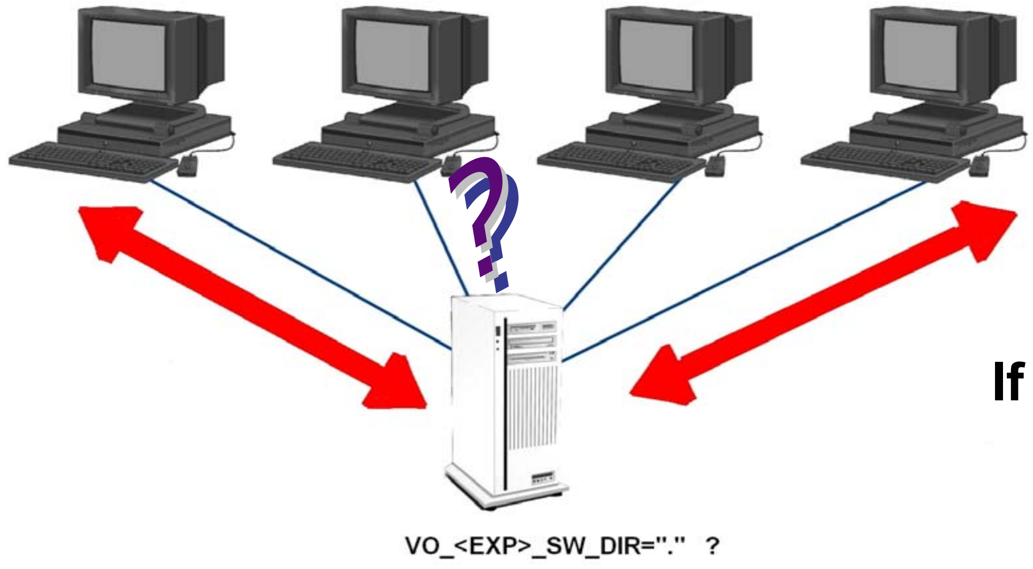
Requirements from site administrators

- No daemon running on the WNs: the WN belongs to the site and not to the GRID
- No root privileges for VO software installation on WN
- Traceable access to the site (no pool accounts)
- In/Outbound connectivity to be avoided
- Site policies applied for external action triggering.
- Strong authentication required
- No shared file system area should be assumed to serve the experiment software area at the site.

Requirements from the Experiment

- Only special users are allowed to install software at a site
- Management of simultaneous and concurrent installations should be in place
- Installation/validation/removal process failure recovery
- Publication of relevant information for the software.
- Installation/validation/removal of software should be done not only through a Grid Job but also via a direct service request in order to assure the right precedence with respect to “normal” jobs
- Validation of the software installed could be done in different steps and at different times with respect to the installation.
- It is up to the experiment to provide scripts for installation/validation/removal of software.

Current Mechanism: The VO_<EXP>SW_AREA

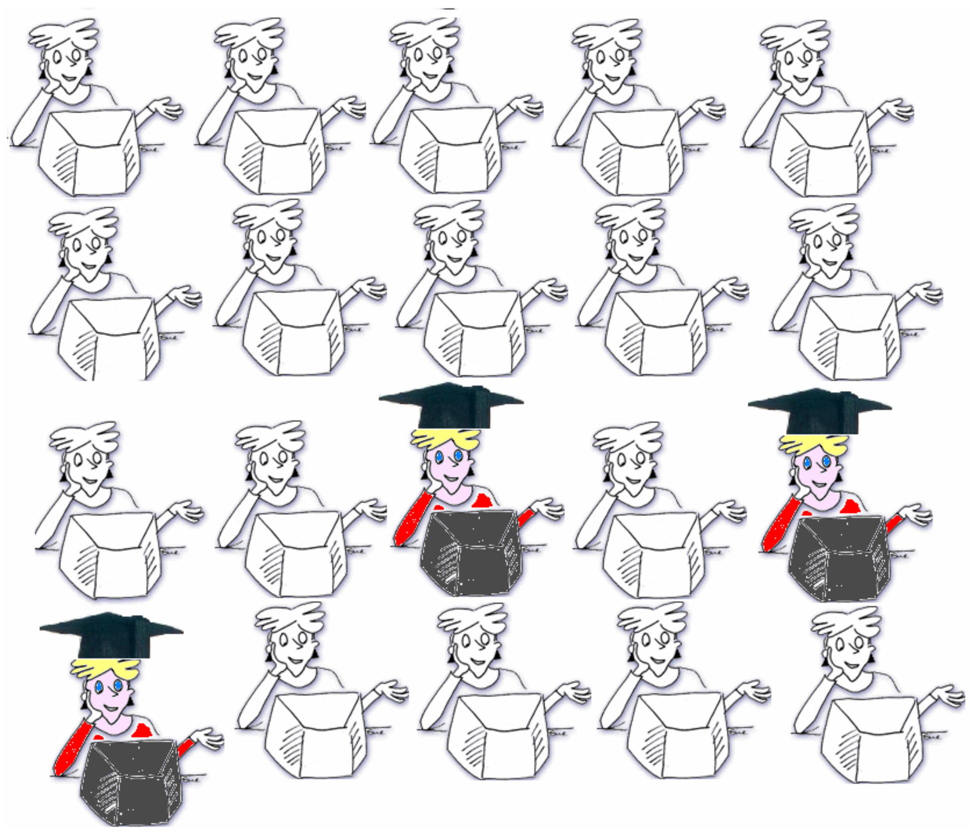


Sites can choose between providing VO space in a shared file system or allow for local software installation on a WN. This is indicated through *the local environment variable* *VO_<EXP>_SW_DIR*.

If it is equal to "." (*meaning in linux the current path*) it means there is no file system shared among WNs and everything is installed locally, on the working directory, and later on scratched by the local batch system.

Vice versa the software can be installed permanently on the shared area and made always accessible by all WNs.

Current Mechanism: The ESM



Only few people in a given collaboration are eligible to install software on behalf of the VO. These special users have write access to the VO space (in case of shared file system) and privileges to publish the tag in the Information System. The VO Manager creates the software administrator Group. People belonging to this group are mapped to a special Administrative VO account in the local grid-mapfile at all sites.

Current Mechanism: The workflow

1. The ESM checks the resources available and decides where the new release must be installed.
2. The ESM creates a tarball file containing scripts to install (and validate later on) and bundles of experiment software.
3. The ESM runs the “special lcg-tools” to install software at a site by submitting “special jobs” explicitly to these sites.
4. The ESM runs verification steps by submitting again other “special jobs” .
5. The ESM publishes tag by using “special tools”. These tags indicate the software is available to subsequent running jobs .

Current Mechanism: Step 0- Decide where install

The first step foreseen in this chain of actions is deciding where the software must be installed.. For that the ESM must know which resources are available for this VO.

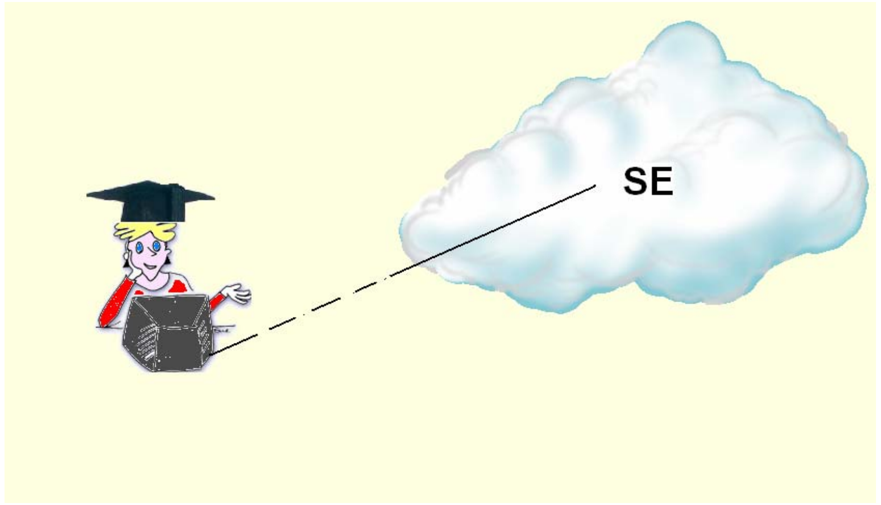
```
% lcg-infosites --vo <your_VO> ce
```

One by one the Computing Elements shown by this command must be considered for subsequent steps. The ESM must also know which Storage Elements are available.

```
% lcg-infosites --vo <your_VO> se
```

The ESM will chose one of these SEs which will be the target of the next command.

Current Mechanism: Step 1-Prepare your tarball



The ESM prepares a stand-alone self-contained distribution of the VO software (all dependencies are solved and needed packages included). The software bundles are packaged together with scripts used to perform the installation and validation).
*Such a bundle is then uploaded to a SE in GRID through the Replica Manager with the –l option. The **lfn** is the name of the tarball without its suffix (tar.gz).*

The name convention for the tarball file is the following:
<vo>-<name_of_SW>-<version>-<patch_level>.tar.gz

```
% tar czvf <vo>-<nameSW>-<Version>-<Release>.tar.gz install_tool\
       install_sw validation_sw <all_packages_constituting_exp_soft >

% lcg-cr --vo <your_VO> file://`pwd`/<vo>-<nameSW>-<Version>-
<Release>.tar.gz -d <SE> /
       -l lfn:<vo>-<nameSW>-<Version>-<Release>.tar.gz
```

Current Mechanism: Step 2: create your jdl for installation and/or validation

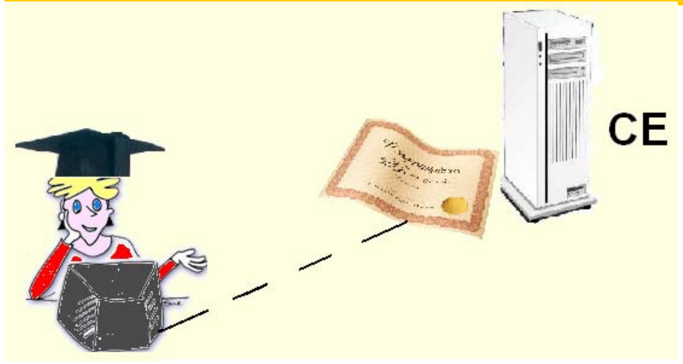
The ESM makes a JDL file in order to trigger an installation/validation process on a given site. Two scenarios:

1. *WNs sharing software installation directory : the software is installed permanently and always accessible from any WN for subsequent jobs.*
2. *Standalone disk space:*
   - *Software is installed and verified in the job working directory in the same job. All dependencies are verified.*
   - *Tag published means that the software can be successfully installed on a WN*
   - *Working dir scratched when job ends*

mandatory **:** ***The fields between <> in the following JDL must be the same as the ones used for naming the tarball***

```
Executable = "/opt/lcg/bin/lcg-ManageSoftware";
OutputSandbox = {"stdout", "stderr"};
stdoutput = "stdout";
stderror = "stderr";
Arguments ="—validate –vo <vo> -V <version> -S <name_of_SW> -N 1 –R\
                    <release>"
Requirements = other.GlueCEUniqueID == "<CE-uniqueID>";
```

Current Mechanism: Step 3 Run the job onto the “certified” CE



For sites on which the installation/certification has been successfully run, the ESM adds (through the appropriate tool) in the IS a Tag that indicates the version of the VO software that has successfully passed the tests.

***Note. The flag published in the IS of the CE is always formed as follows: <flag>==VO-<vo>-<name_of_SW>-<version>-<release>. If the user specifies explicitly the flag with –flag option in the validation JDL , VO-<vo> will be in any case added at the head of the string. (<flag>== VO-<vo>*-<flag user provided>)**

```
Executable = “/opt/lcg/bin/lcg-ManageSoftware”;
InputSandbox = “<name_of_run_script>”
OutputSandbox = {“stdout”, “stderr”};
stdoutput = “stdout”;
stderror = “stderr”;
Arguments =“—run –vo <vo> -V <version> -S <name_of_SW> -N 1 –R\
          <release> --run_script <name_of_run_script>”
Requirements = \
Member(“<flag>”,other.GlueHostApplicationSoftwareRunTimeEnvironment);
```

Lcg-ManageSoftware and lcg-ManageVOTag

Lcg-ManageSoftware allows the ESM to automatically install/validate/run and de-install a version of experiment software. At the end of the process, if everything went OK it publishes (removes) from the IS the corresponding flag using lcg-ManageVOTag command

lcg-ManageVOTag: is the command which adds/list/removes the flags published on the Information System of a given CE.
It could be also used as a standalone application.
It is also invoked by Lcg-manageSoftware

Lcg-ManageSoftware

```
% man lcg-ManageSoftware
```

Mandatory to provide:

1. The action to be chosen among: --install –validate –run –uninstall
2. The –S option (the name of the software)
3. The –V option (a string describing the version number of the software going to be installed
4. The –vo option (the VO you belong)
5. The –R option (the number of release or patch-level, D=1)

Optionally:

-N (the number of tarballs that constitute this release)

--flag (allows the user to define a customized flag (remember that in any case the prefix VO-<vo>- will be added to the head of this string

--CE the computing element (mandatory if the user submits its job with –R option)

--SE the storage element where the user wants download the tarball(s)

--run_script that allow the user to specify a different name than the default (run_sw) for the script to be used in running a given version of software

--ov (D=1) option allows for accomplish the validation with (=1) or without (=0) an preinstallation process on the same job. This is useful for shared file system topology

Lcg-ManageVOTag

```
% man lcg-ManageVOtag
```

Mandatory to provide:

The action

–add to add a new tag in the IS of a given CE

--remove to remove a given tag in the IS of a given CE

--list to list the tags published in a given CE

--clean to remove all tags in a given CE

1. -vo option (the VO you belong)
2. -host the CE where you want perform the action
3. -tag (for –add and –remove actions) : the flag to remove/added in the IS of the specified CE

Hands-on

Hands on

Exercise 1.

The package of the software is a empty-file (example_file)
The **install_tool** scripts do nothing.

The installation script (**install_sw**) will create under the root-installation-dir a directory named <NameOfTheDayOfWeek>-date and copy on this directory the file example_file

The **validation_sw** tool checks for the existence of this file and otherwise performs the installation

The run script **run_sw** which must be included in the InputSandbox checks for the existences of this files and gives us the name of the host otherwise it ends with a non zero result.

The **uninstall_sw** scripts will remove the <NameOfTheDayOfWeek>-date dir.

Hands on

We kindly invite you to:

1. Create a tarball with the following features:

   name of software=tutorial

   version=1.0.0

   release=1

   number_of_tarball=1

It should contain the install_sw and install_tool and validation_sw scripts plus the package of the software.

2. Build a JDL (tutorial.jdl) which install and validate at the same time this package (--validate option) for the vo you belong (gilda) on a CE you chosen
3. See the list of flags published for your VO on this site through lcg-ManageVOTag after this operation.
4. Create a JDL (tutorial_run.jdl) which runs exactly the software previously installed. The run_sw must be included in the InputSandBox field
5. Create a JDL (tutorial_uninstall.jdl) which will remove this package from this CE. It must include in the inputSandbox the script uninstall_sw
6. Check (after 5) the list of flags still published by the CE

Hands on

Exercise 2.

The package of the software is a tarball (user.tar.gz) containing several files with the C++ code and a Makefile to compile it.
The **install_tool** script does nothing**.**

The installation script (**install_sw**) will create under the root-installation-dir a directory named <NameOfTheDayOfWeek>-date, untars the tarball, does a gmake and copy under theroot directory the executable.

The **validation_sw** tool checks for the existence of this file and- otherwise- performs the installation.

The run script **run_my_test** which must be included in the InputSandbox, checks for the existence of the executable and runs it with options user defined.

Hands on

We kindly invite you to:

1. Create an installation tarball file with the following features:

   name of software=tutorial

   version=2.0.0

   release=1

   number_of_tarball=1

It should contain the install_sw and install_tool and validation_sw scripts plus the package of the software (user.tar.gz).

2. Build a JDL (tutorial.jdl) which install and validate at the same time this package (--validate option) for the vo you belong (gilda) on a CE you chosen and publish a user-defined flag
3. See the list of flags published for your VO on this site through lcg-ManageVOTag after this operation.
4. Create a JDL (tutorial_run.jdl) which runs exactly the software previously installed. A different name for the run script must be supplied and shipped with the sandbox.

   PS. In building this script the user can specify the options for call the executable among the ones specified in the README file.