LHC Computing Grid

LCG - TAR_UI - Generic Configuration Reference

Document identifier:	**LCG-GIS-CR-TAR_UI**
EDMS id:	**none**
Version:	
Date:	**January 16, 2006**
Section:	**LCG Grid Infrastructure Support**
Document status:	**ACTIVE**
Author(s):	Keeble, Oliver;LCG Deployment - GIS team;Retico,Antonio;Vidic,Valentin;
File:	**TAR_UI**

Abstract: *Configuration steps done by the YAIM script 'configure_TAR_UI'*

CONTENTS

1. INTRODUCTION 4
2. VARIABLES 5
3. SET TAR UI DEFAULTS 7
   3.1. SPECIFICATION OF FUNCTION: CONFIG_TAR_UI_DEFAULTS 7
4. CHECK TAR INSTALLATION 8
   4.1. SPECIFICATION OF FUNCTION: CONFIG_CHECK_TAR 8
5. INSTALL CA CERTIFICATES 9
   5.1. SPECIFICATION OF FUNCTION: INSTALL_CERTS_USERLAND 9
6. FIX CRL UPDATE SCRIPT 10
   6.1. SPECIFICATION OF FUNCTION: CONFIG_FIX_EDG-FETCH-CRL-CRON 10
7. SET-UP UPDATING OF CRLS 11
   7.1. SPECIFICATION OF FUNCTION: CONFIG_CRL 11
8. SET-UP REPLICA MANAGER 12
   8.1. SPECIFICATION OF FUNCTION: CONFIG_REPLICA_MANAGER 12
9. SET-UP USER ENVIRONMENT 13
   9.1. SPECIFICATION OF FUNCTION: CONFIG_TAR_USER_ENV 13
10. SET-UP LCG ENVIRONMENT VARIABLES 14
   10.1. SPECIFICATION OF FUNCTION: CONFIG_LCGENV 16
11. SET-UP JAVA LOCATION 17
   11.1. SPECIFICATION OF FUNCTION: CONFIG_JAVA 17
12. SET-UP R-GMA CLIENT 18
   12.1. SPECIFICATION OF FUNCTION: CONFIG_RGMA_CLIENT 18
13. SET-UP WORKLOAD MANAGER CLIENT 19
   13.1. SPECIFICATION OF FUNCTION: CONFIG_WORKLOAD_MANAGER_CLIENT 20
14. CONFIGURE THE VOMSES FILES 21
   14.1. SPECIFICATION OF FUNCTION: CONFIG_VOMSES 21
15. SET-UP FILE TRANSFER SERVICE CLIENT 22
   15.1. SPECIFICATION OF FUNCTION: CONFIG_FTS_CLIENT 22

16. SOURCE CODE 23
16.1. CONFIG_TAR_UI_DEFAULTS 23
16.2. CONFIG_CHECK_TAR 23
16.3. INSTALL_CERTS_USERLAND 24
16.4. CONFIG_FIX_EDG-FETCH-CRL-CRON 25
16.5. CONFIG_CRL 25
16.6. CONFIG_REPLICA_MANAGER 26
16.7. CONFIG_TAR_USER_ENV 27
16.8. CONFIG_LCGENV 30
16.9. CONFIG_JAVA 34
16.10. CONFIG_RGMA_CLIENT 36
16.11. CONFIG_WORKLOAD_MANAGER_CLIENT 37
16.12. CONFIG_VOMSES 39
16.13. CONFIG_FTS_CLIENT 40

1. INTRODUCTION

This document lists the manual steps for the installation and configuration of a LCG TAR_UI Node. Furthermore it provides a specification of the YAIM functions used to configure the node with the script-based configuration.

The configuration has been tested on a standard Scientific Linux 3.0 Installation.

Link to this document:
This document is available on the *Grid Deployment* web site

```
http://www.cern.ch/grid-deployment/gis/lcg-GCR/index.html
```

2. VARIABLES

In order to set-up a TAR_UI node, you need at least the following variables to be correctly configured in the site configuration file (site-info.def):

BDII_HOST : BDII Hostname.

CE_HOST : Computing Element Hostname.

DPM_HOST : Host name of the DPM host, used also as a default DPM for the lcg-stdout-mon .

EDG_WL_SCRATCH : Optional scratch directory for jobs.

EDG_WL_SCRATCH : Optional scratch directory for jobs.

FTS_SERVER_URL : URL of the File Transfer Service server.

GLOBUS_TCP_PORT_RANGE : Port range for Globus IO.

GSSKLOG : yes or no, indicating whether the site provides an AFS authentication server which maps gsi credentials into Kerberos tokens .

GSSKLOG_SERVER : If GSSKLOG is yes, the name of the AFS authentication server host.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

JAVA_LOCATION : Path to Java VM installation. It can be used in order to run a different version of java installed locally.

MON_HOST : MON Box Hostname.

OUTPUT_STORAGE : Default Output directory for the jobs.

PX_HOST : PX hostname.

RB_HOST : Resource Broker Hostname.

REG_HOST : RGMA Registry hostname.

SE_LIST : A list of hostnames of the SEs available at your site.

SITE_NAME : Your GIIS.

VOBOX_HOST : VOBOX hostname.

VOS : List of supported VOs. For each item listed in the VOS variable you need to create a set of new variables as follows:

> **VO_<VO-NAME>_SE :** Default SE used by the VO. WARNING: VO-NAME must be in capital cases.

VO_<VO-NAME>_SW_DIR : Area on the WN for the installation of the experiment software. If on the WNs a predefined shared area has been mounted where VO managers can pre-install software, then these variable should point to this area. If instead there is not a shared area and each job must install the software, then this variables should contain a dot (.).Anyway the mounting of shared areas, as well as the local installation of VO software is not managed by *yaim* and should be handled locally by Site Administrators. WARNING: VO-NAME must be in capital cases.

VO_<VO-NAME>_VOMSES : List of entries for the vomses files for this VO. Multiple values can be given if enclosed in single quotes.

3. SET TAR UI DEFAULTS

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_tar_ui_defaults*'.

Defaults for various services are set (if not already defined in *site-info.def*):

```
RB_HOST=lxn1177.cern.ch
PX_HOST=my-proxy.cern.ch
BDII_HOST=lcg-bdii.cern.ch
MON_HOST=norgma
REG_HOST=norgma
VOS="atlas alice lhcb cms dteam sixt na48"
```

3.1. SPECIFICATION OF FUNCTION: CONFIG_TAR_UI_DEFAULTS

The function *'config_tar_ui_defaults'* needs the following variables to be set in the configuration file:

BDII_HOST : BDII Hostname.

MON_HOST : MON Box Hostname.

PX_HOST : PX hostname.

RB_HOST : Resource Broker Hostname.

REG_HOST : RGMA Registry hostname.

VOS : List of supported VOs.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_tar_ui_defaults
```

The code is also reproduced in 16.1..

4. CHECK TAR INSTALLATION

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function ’*config_check_tar*’.

Check if <*INSTALL_ROOT*> directory exists.

Check if dependency software is installed (test if <*INSTALL_ROOT*>*/edg/share/java/log4j.jar* exists).

4.1. SPECIFICATION OF FUNCTION: CONFIG_CHECK_TAR

The function *’config_check_tar’* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_check_tar
```

The code is also reproduced in 16.2..

5. INSTALL CA CERTIFICATES

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*install_certs_userland*'.

If CA certificates are not already installed in */etc/grid-security/certificates*, certificate RPMs are downloaded from *http://grid-deployment.web.cern.ch/grid-deployment/download/RpmDir/security/index.html* and extracted to *<INSTALL_ROOT>/etc/grid-security/certificates*.

5.1. SPECIFICATION OF FUNCTION: INSTALL_CERTS_USERLAND

The function *'install_certs_userland'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/install_certs_userland
```

The code is also reproduced in 16.3..

6. Fix CRL update script

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_fix_edg-fetch-crl-cron*'.

CRL update script (*<INSTALL_ROOT>/edg/etc/cron/edg-fetch-crl-cron*) is modified to use *<INSTALL_ROOT>/edg* as *EDG_LOCATION* and *<INSTALL_ROOT>/etc/grid-security/certificates* as *X509_CERT_DIR*.

6.1. Specification of function: config_fix_edg-fetch-crl-cron

The function '*config_fix_edg-fetch-crl-cron*' needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_fix_edg-fetch-crl-cron
```

The code is also reproduced in 16.4..

7. Set-up updating of CRLs

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function *'config_crl'*.

Cron script is installed to fetch new versions of CRLs four times a day. The time when the script is run is randomized in order to distribute the load on CRL servers. If the configuration is run as root, the cron entry is installed in */etc/cron.d/edg-fetch-crl*, otherwise it is installed as a user cron entry.

CRLs are also updated immediately by running the update script (*<INSTALL_ROOT>/edg/etc/cron/edg-fetch-crl-cron*).

Logrotate script is installed as */etc/logrotate.d/edg-fetch-crl* to prevent the logs from growing indefinitely.

7.1. Specification of function: config_crl

The function *'config_crl'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_crl
```

The code is reproduced also in 16.5..

8. Set-up Replica Manager

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_replica_manager*'.

Variable substitutions are generated in *<INSTALL_ROOT>/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local*:

```
@EDG.LOCATION@|<INSTALL_ROOT>/edg|location of edg middleware
@LOCALDOMAIN@|<domain>|the local domain
@DEFAULT.SE@|<SE_HOST>|the host of the close SE
@DEFAULT.CE@|<CE_HOST>|the host of the close CE
@INFOSERVICE@|MDS|The info provider to use. It can be Stub, MDS or RGMA
@RLS.MODE@|LrcOnly|The mode the RLS should be run in. LrcOnly or WithRli
@STUBFILE@||The properties file for the static file - only needed in Stub mode
@MDS.HOST@|<BDII_HOST>|The host of the MDS info provider
@MDS.PORT@|2170|The port of the MDS info provider
@ROS.FAILURE@|false|Fail if no ROS is available
@CONF.GCC@|_gcc3_2_2|The gcc suffix as used on the build box (empty for 2.95, _gcc3_2_2 for 3.2.)
@IGNORE.PREFIX@|true|Whether the RM will ignore the lfn and guid prefix.
@GRIDFTP.DCAU@|false|Does GridFTP use Data Channel Authentication (DCAU)
@GRIDFTP.STREAMS.SMALL@|1|The default number of stream to use for a small file
@GRIDFTP.STREAMS.BIG@|3|The default number of stream to use for a big file
@GRIDFTP.FILESIZE.THRESHOLD@|100|The Threshold (in MB) above which a file to transfer is considered "big"
```

The value of <domain> is determined by running *hostname -d*. Using these substitutions and templates in *<INSTALL_ROOT>/edg/etc/edg-replica-manager/*, Replica Manager is configured by generating files in *<EDG_LOCATION>/var/etc/edg-replica-manager*:

```
<INSTALL_ROOT>/edg/sbin/edg-replica-manager-configure <INSTALL_ROOT>/edg/etc/edg-replica-manager/edg-replica-manage
```

8.1. Specification of function: config_replica_manager

The function *'config_replica_manager'* needs the following variables to be set in the configuration file:

BDII_HOST : BDII Hostname.

CE_HOST : Computing Element Hostname.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

SE_LIST : A list of hostnames of the SEs available at your site.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_replica_manager
```

The code is also reproduced in 16.6..

9. Set-up user environment

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_tar_user_env*'.

<INSTALL_ROOT>/etc/profile.d/grid_env.sh and *<INSTALL_ROOT>/etc/profile.d/grid_env.csh* are fixed to use the correct location of LCG software:

```
UI_LOC=<INSTALL_ROOT>
```

If CA certificates are installed in *<INSTALL_ROOT>/etc/grid-security/certificates*, *X509_CERT_DIR* is also modified appropriately.

9.1. Specification of function: config_tar_user_env

The function '*config_tar_user_env*' needs the following variables to be set in the configuration file:

GLOBUS_TCP_PORT_RANGE : Port range for Globus IO.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_tar_user_env
```

The code is also reproduced in 16.7..

10. Set-up LCG environment variables

Author(s): Retico,Antonio
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_lcgenv*'.

The LCG middleware needs some environment variables to be set up at boot time. The variable should be available both in 'bash-like' shells and in 'csh-like' shells.

This can be obtained in different ways:

The simplest way, if you have 'root' permissions, is to put a shell script for each of the supported shell 'families' in the directory */etc/profile.d*. The script will be automatically sourced at start up.

If instead you are not a superuser and you are doing the installation in a private directory (e.g. you are installing a Re-locatable Distribution of a Worker Node or a User Interface in the *<INSTALL_ROOT>* directory), you could create the scripts in the directory *<INSTALL_ROOT>/etc/profile.d*, in order to have the variables automatically set up by LCG tools.

The list of the environment variables to be set up follows:

LCG_GFAL_INFOSYS: Hostname of the BDII node.

MYPROXY_SERVER: Hostname of the Proxy server.

PATH: Add to the PATH variable the path */opt/d-cache-client/bin*

LD_LIBRARY_PATH: Add to the LD_LIBRARY_PATH variable the path */opt/d-cache-client/dcap*

SRM_PATH: Installation directory of the srm client. The default value for this variable is */opt/d-cache-client/srm*

VO_<VO-NAME>_SW_DIR: For each virtual organization <VO-NAME> An environment variable VO_<VO-NAME>_SW_DIR is needed. This variable points to the installation directory of the VO software.

VO_<VO-NAME>_DEFAULT_SE: For each virtual organization <VO-NAME> An environment variable VO_<VO-NAME>_DEFAULT_SE is needed. This variable points to the Default Storage Element for that VO.

The examples given hereafter refer to the simple configuration method described above. In the following description we will refer to the two possible locations as to the <LCG_ENV_LOC>.
So, according to the cases above described, either

<LCG_ENV_LOC>=/etc/profile.d

or

<LCG_ENV_LOC>=<INSTALL_ROOT>/etc/profile.d

Examples:

- Example of file *<LCG_ENV_LOC>/lcgenv.sh*:

```
#!/bin/sh
export LCG_GFAL_INFOSYS=lxb1769.cern.ch:2170
export MYPROXY_SERVER=lxb1774.cern.ch
export PATH="${PATH}:/opt/d-cache-client/bin"
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/opt/d-cache-client/dcap
export SRM_PATH=/opt/d-cache-client/srm
export VO_ATLAS_SW_DIR=lxb1780.cern.ch
export VO_ATLAS_DEFAULT_SE=lxb1780.cern.ch
export VO_ALICE_SW_DIR=lxb1780.cern.ch
export VO_ALICE_DEFAULT_SE=lxb1780.cern.ch
export VO_LHCB_SW_DIR=lxb1780.cern.ch
export VO_LHCB_DEFAULT_SE=lxb1780.cern.ch
export VO_CMS_SW_DIR=lxb1780.cern.ch
export VO_CMS_DEFAULT_SE=lxb1780.cern.ch
export VO_DTEAM_SW_DIR=lxb1780.cern.ch
export VO_DTEAM_DEFAULT_SE=lxb1780.cern.ch
```

- Example of file *<LCG_ENV_LOC>/lcgenv.csh*:

```
#!/bin/csh
setenv LCG_GFAL_INFOSYS lxb1769.cern.ch:2170
setenv MYPROXY_SERVER lxb1774.cern.ch
setenv PATH "${PATH}:/opt/d-cache-client/bin"
setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:/opt/d-cache-client/dcap
setenv SRM_PATH /opt/d-cache-client/srm
setenv VO_ATLAS_SW_DIR lxb1780.cern.ch
setenv VO_ATLAS_DEFAULT_SE lxb1780.cern.ch
setenv VO_ALICE_SW_DIR lxb1780.cern.ch
setenv VO_ALICE_DEFAULT_SE lxb1780.cern.ch
setenv VO_LHCB_SW_DIR lxb1780.cern.ch
setenv VO_LHCB_DEFAULT_SE lxb1780.cern.ch
setenv VO_CMS_SW_DIR lxb1780.cern.ch
setenv VO_CMS_DEFAULT_SE lxb1780.cern.ch
setenv VO_DTEAM_SW_DIR lxb1780.cern.ch
setenv VO_DTEAM_DEFAULT_SE lxb1780.cern.ch
```

WARNING: The two scripts must be executable by all users.

```
> chmod a+x ${LCG_ENV_LOC}/lcgenv.csh
> chmod a+x ${LCG_ENV_LOC}/lcgenv.sh
```

10.1. Specification of function: config_lcgenv

The function *'config_lcgenv'* needs the following variables to be set in the configuration file:

BDII_HOST : BDII Hostname.

CE_HOST : Computing Element Hostname.

DPM_HOST : Host name of the DPM host, used also as a default DPM for the lcg-stdout-mon .

EDG_WL_SCRATCH : Set this if you want jobs to use a particular scratch area.

EDG_WL_SCRATCH : Set this if you want jobs to use a particular scratch area.

GLOBUS_TCP_PORT_RANGE : Port range for Globus IO.

GSSKLOG : yes or no, indicating whether the site provides an AFS authentication server which maps gsi credentials into Kerberos tokens .

GSSKLOG_SERVER : If GSSKLOG is yes, the name of the AFS authentication server host.

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

PX_HOST : PX hostname.

SE_LIST : A list of hostnames of the SEs available at your site.

SITE_NAME : Your GIIS.

VOBOX_HOST : VOBOX hostname.

VOS : List of supported VOs. For each item listed in the VOS variable you need to create a set of new variables as follows:

> **VO_<VO-NAME>_SE :** Default SE used by the VO. WARNING: VO-NAME must be in capital cases.
>
> **VO_<VO-NAME>_SW_DIR :** Area on the WN for the installation of the experiment software. If on the WNs a predefined shared area has been mounted where VO managers can pre-install software, then these variable should point to this area. If instead there is not a shared area and each job must install the software, then this variables should contain a dot (.).Anyway the mounting of shared areas, as well as the local installation of VO software is not managed by *yaim* and should be handled locally by Site Administrators. WARNING: VO-NAME must be in capital cases.

The function does exit with return code 1 if they are not set.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_lcgenv
```

The code is reproduced also in 16.8..

11. SET-UP JAVA LOCATION

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_java*'.

Since Java is not included in the LCG distribution, Java location needs to be configured with yaim.

If <JAVA_LOCATION> is not defined in *site-info.def*, it is determined from installed Java RPMs (if available).

In relocatable distribution, JAVA_HOME environment variable is defined in *<INSTALL_ROOT>/etc/profile.d/grid_en* and *<INSTALL_ROOT>/etc/profile.d/grid_env.csh*.

Otherwise, JAVA_HOME is defined in */etc/java/java.conf* and */etc/java.conf* and Java binaries added to PATH in *<INSTALL_ROOT>/edg/etc/profile.d/j2.sh* and *<INSTALL_ROOT>/edg/etc/profile.d/j2.csh*.

11.1. SPECIFICATION OF FUNCTION: CONFIG_JAVA

The function *'config_java'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

JAVA_LOCATION : Path to Java VM installation. It can be used in order to run a different version of java installed locally.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_java
```

The code is reproduced also in 16.9..

12. SET-UP R-GMA CLIENT

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_rgma_client*'.

R-GMA client configuration is generated in *<INSTALL_ROOT>/glite/etc/rgma/rgma.conf* by running:

```
<INSTALL_ROOT>/glite/share/rgma/scripts/rgma-setup.py --secure=no --server=<MON_HOST> --registry=<REG_HOST> --schem
```

<INSTALL_ROOT>/edg/etc/profile.d/edg-rgma-env.sh and *<INSTALL_ROOT>/edg/etc/profile.d/edg-rgma-env.csh* with the following functionality:

- RGME_HOME is set to *<INSTALL_ROOT>/glite*
- APEL_HOME is set to *<INSTALL_ROOT>/glite*
- *<INSTALL_ROOT>/glite/lib/python* is added to PYTHONPATH
- *<INSTALL_ROOT>/glite/lib* is added to LD_LIBRARY_PATH.

These files are sourced into the users environment from *<INSTALL_ROOT>/etc/profile.d/z_edg_profile.sh* and *<INSTALL_ROOT>/etc/profile.d/z_edg_profile.csh*.

12.1. SPECIFICATION OF FUNCTION: CONFIG_RGMA_CLIENT

The function '*config_rgma_client*' needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

MON_HOST : MON Box Hostname.

REG_HOST : RGMA Registry hostname.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_rgma_client
```

The code is also reproduced in 16.10..

13. SET-UP WORKLOAD MANAGER CLIENT

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function ’*config_workload_manager_client*’.

For each of the supported VOs, a configuration file is generated in *<INSTALL_ROOT>/edg/etc/<vo>/edg_wl_ui.conf*:

```
[
VirtualOrganisation = "<vo>";
NSAddresses = "<RB_HOST>:7772";
LBAddresses = "<RB_HOST>:9000";
## HLR location is optional. Uncomment and fill correctly for
## enabling accounting
#HLRLocation = "fake HLR Location"
## MyProxyServer is optional. Uncomment and fill correctly for
## enabling proxy renewal. This field should be set equal to
## MYPROXY_SERVER environment variable
MyProxyServer = "<PX_HOST>"
]
```

This file defines RB and MyProxy server to be used for a specific VO.

Defaults common to all VOs are defined in *<INSTALL_ROOT>/edg/etc/edg_wl_ui_cmd_var.conf*:

```
[
rank = - other.GlueCEStateEstimatedResponseTime;
requirements = other.GlueCEStateStatus == "Production";
RetryCount = 3;
ErrorStorage = "/tmp";
OutputStorage = "<OUTPUT_STORAGE>";
ListenerPort = 44000;
ListenerStorage = "/tmp";
LoggingTimeout = 30;
LoggingSyncTimeout = 30;
LoggingDestination = "<RB_HOST>:9002";
# Default NS logger level is set to 0 (null)
# max value is 6 (very ugly)
NSLoggerLevel = 0;
DefaultLogInfoLevel = 0;
DefaultStatusLevel = 0;
DefaultVo = "unspecified";
]
```

and *<INSTALL_ROOT>/edg/etc/edg_wl_ui_gui_var.conf*:

```
[
JDLEDefaultSchema = "Glue";
Glue = [
rank = - other.GlueCEStateEstimatedResponseTime;
rankMPI = other.GlueCEStateFreeCPUs;
requirements = other.GlueCEStateStatus == "Production"
];
EDG = [
rank = - other.EstimatedTraversalTime;
rankMPI = other.FreeCPUs;
requirements = true
];
RetryCount = 3;
ErrorStorage = "/tmp";
OutputStorage = "<OUTPUT_STORAGE>;
ListenerPort = 44000;
ListenerStorage = "/tmp";
LoggingTimeout = 30;
LoggingSyncTimeout = 30;
LoggingDestination = "<RB_HOST>:9002";
# Default NS logger level is set to 0 (null)
# max value is 6 (very ugly)
NSLoggerLevel = 0;
DefaultLogInfoLevel = 0;
DefaultStatusLevel = 0;
DefaultVo = "unspecified";
]
```

Directory for storing job output sandboxes (<OUTPUT_STORAGE>) is created.

Finally, *edg-wl-ui-env.csh*, *edg-wl-ui-env.sh*, *edg-wl-ui-gui-env.csh* and *edg-wl-ui-gui-env.sh* are copied from *<INSTALL_ROOT>/edg/etc/profile.d/* to *<INSTALL_ROOT>/edg/var/etc/profile.d*. These scrips are sourced from */etc/profile.d* and are used to setup the environment for EDG programs.

13.1. SPECIFICATION OF FUNCTION: CONFIG_WORKLOAD_MANAGER_CLIENT

The function *'config_workload_manager_client'* needs the following variables to be set in the configuration file:

INSTALL_ROOT : Installation root - change if using the re-locatable distribution.

OUTPUT_STORAGE : Default Output directory for the jobs.

PX_HOST : PX hostname.

RB_HOST : Resource Broker Hostname.

VOS : List of supported VOs.

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_workload_manager_client
```

The code is also reproduced in 16.11..

14. Configure the VOMSES files

Author(s): Keeble, Oliver
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the *yaim* function '*config_vomses*'.

This function creates the vomses configuration files. Each file contains one line of information describing a VOMS server. This information is used by commands like voms-proxy-init in order to create a voms proxy. The necesssary information is taken from parameters in site-info.def (VO$_X XX_V OMSES$). The files are placed in central directories if run as root, or in *.edg/vomses if run as a user.*

14.1. Specification of function: config_vomses

The function 'config_vomses' *needs the following variables to be set in the configuration file:*

INSTALL_ROOT : *Installation root - change if using the re-locatable distribution.*

VOS : *List of supported VOs. For each item listed in the VOS variable you need to create a set of new variables as follows:*

- **VO_<VO-NAME>_VOMSES :** *List of entries for the vomses files for this VO. Multiple values can be given if enclosed in single quotes.*

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_vomses
```

The code is also reproduced in 16.12..

15. Set-up File Transfer Service client

Author(s): Vidic,Valentin
Email : support-lcg-manual-install@cern.ch

This chapter describes the configuration steps done by the yaim *function* 'config_fts_client'.

If <FTS_SERVER_URL> is set, gLite FTS client is configured by creating <INSTALL_ROOT>/glite/etc/services.xml

```
<services>

  <service name="EGEEfts">
    <parameters>
      <endpoint><FTS_SERVER_URL>/services/FileTransfer</endpoint>
      <type>org.glite.FileTransfer</type>
      <version>3.0.0</version>
    </parameters>
  </service>

  <service name="EGEEchannel">
    <parameters>
      <endpoint><FTS_SERVER_URL>/services/ChannelManagement</endpoint>
      <type>org.glite.ChannelManagement</type>
      <version>3.0.0</version>
    </parameters>
  </service>

</services>
```

15.1. Specification of function: config_fts_client

The function 'config_fts_client' *needs the following variables to be set in the configuration file:*

FTS_SERVER_URL : *URL of the File Transfer Service server.*

INSTALL_ROOT : *Installation root - change if using the re-locatable distribution.*

The original code of the function can be found in:

```
/opt/lcg/yaim/functions/config_fts_client
```

The code is also reproduced in 16.13..

16. Source code

16.1. CONFIG_TAR_UI_DEFAULTS

```
function config_tar_ui_defaults () {


if [ -z "$RB_HOST" ]; then
    RB_HOST=gdrb01.cern.ch
    PX_HOST=my-proxy.cern.ch
fi

if [ -z "$BDII_HOST" ]; then
    BDII_HOST=lcg-bdii.cern.ch
fi

if [ -z "$MON_HOST" ]; then
    MON_HOST="norgma"
    REG_HOST="norgma"
fi

if [ -z "$VOS" ]; then
    VOS="atlas alice lhcb cms dteam sixt na48"
fi

return 0

}
```

16.2. CONFIG_CHECK_TAR

```
function config_check_tar () {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if [ ! -d ${INSTALL_ROOT} ]; then
        echo "please make the distribution available under ${INSTALL_ROOT}"
        echo "or update your site-info.def file to reflect the correct path"
        return 1
fi

if [ $UID -ne 0 ]; then
        if [ ! -f ${INSTALL_ROOT}/edg/share/java/log4j.jar ]; then
                echo "You are attempting a userland installation"
                echo "Have you untarred the dependency software into"
                echo "${INSTALL_ROOT}/edg yet?"
                return 1
        fi
fi

return 0
}
```

16.3. INSTALL_CERTS_USERLAND

```
function install_certs_userland () {

if central_certs; then
    echo "Certificates found in /etc/grid-security/certificates"
    echo "Not installing relocated versions"
    return 0
fi

CA_WGET="http://grid-deployment.web.cern.ch/grid-deployment/download/RpmDir/security/index.html"

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if ! ( which rpm2cpio > /dev/null 2>&1 ); then
echo "rpm2cpio is required to install the CA rpms"
return 1
fi

mkdir tmp.$$ || return 1

cd tmp.$$


wget -l1 -nd -r --quiet $CA_WGET

if ! (ls *.rpm > /dev/null 2>&1); then
echo "Couldn't download the CA rpms"
return 1
fi

for i in *.rpm; do
rpm2cpio $i | cpio -i --make-directories --quiet
done

if [ ! -d ${INSTALL_ROOT}/etc/grid-security/certificates ]; then
mkdir -p ${INSTALL_ROOT}/etc/grid-security/certificates
fi

mv -f etc/grid-security/certificates/* ${INSTALL_ROOT}/etc/grid-security/certificates

cd - > /dev/null

rm -rf tmp.$$

return 0

}
```

16.4. CONFIG_FIX_EDG-FETCH-CRL-CRON

```
function config_fix_edg-fetch-crl-cron () {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

ex $INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron <<EOF
%s#^\$para{EDG_LOCATION}=".*";#\$para{EDG_LOCATION}="$INSTALL_ROOT/edg";#
w
q
EOF

ex $INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron <<EOF
%s#^\$para{X509_CERT_DIR}=".*";#\$para{X509_CERT_DIR}="$INSTALL_ROOT/etc/grid-security/certificates";#
w
q
EOF

return 0

}
```

16.5. CONFIG_CRL

```
config_crl(){

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

let minute="$RANDOM%60"

let h1="$RANDOM%24"
let h2="($h1+6)%24"
let h3="($h1+12)%24"
let h4="($h1+18)%24"

if !( echo "${NODE_TYPE_LIST}" | grep TAR > /dev/null ); then

    if [ ! -f /etc/cron.d/edg-fetch-crl ]; then
echo "Now updating the CRLs - this may take a few minutes..."
$INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron >> /var/log/edg-fetch-crl-cron.log 2>&1
    fi

cron_job edg-fetch-crl root "$minute $h1,$h2,$h3,$h4 * * * $INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron >> /var/lc

    cat <<EOF > /etc/logrotate.d/edg-fetch
/var/log/edg-fetch-crl-cron.log  {
    compress
    monthly
    rotate 12
    missingok
    ifempty
    create
}
EOF

else

    cron_job edg-fetch-crl `whoami` "$minute $h1,$h2,$h3,$h4 * * * $INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron >>
    if [ ! -d $INSTALL_ROOT/edg/var/log ]; then
mkdir -p $INSTALL_ROOT/edg/var/log
    fi
    echo "Now updating the CRLs - this may take a few minutes..."
    $INSTALL_ROOT/edg/etc/cron/edg-fetch-crl-cron >> $INSTALL_ROOT/edg/var/log/edg-fetch-crl-cron.log 2>&1

fi

return 0
}
```

16.6. CONFIG_REPLICA_MANAGER

```
config_replica_manager(){

# SE_HOST and CE_HOST are not strictly required
requires BDII_HOST

se_host="${SE_LIST%% *}"

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if [ -f ${INSTALL_ROOT}/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local ]; then
        mv -f ${INSTALL_ROOT}/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local /tmp/edg-replica-ma
fi

cat <<EOF > ${INSTALL_ROOT}/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local
@EDG.LOCATION@|$INSTALL_ROOT/edg|location of edg middleware
@LOCALDOMAIN@|`hostname -d`|the local domain
@DEFAULT.SE@|$se_host|the host of the close SE
@DEFAULT.CE@|$CE_HOST|the host of the close CE
@INFOSERVICE@|MDS|The info provider to use. It can be Stub, MDS or RGMA
@RLS.MODE@|LrcOnly|The mode the RLS should be run in. LrcOnly or WithRli
@STUBFILE@||The properties file for the static file - only needed in Stub mode
@MDS.HOST@|$BDII_HOST|The host of the MDS info provider
@MDS.PORT@|2170|The port of the MDS info provider
@ROS.FAILURE@|false|Fail if no ROS is available
@CONF.GCC@|_gcc3_2_2|The gcc suffix as used on the build box (empty for 2.95, _gcc3_2_2 for 3.2.)
@IGNORE.PREFIX@|true|Whether the RM will ignore the lfn and guid prefix.
@GRIDFTP.DCAU@|false|Does GridFTP use Data Channel Authentication (DCAU)
@GRIDFTP.STREAMS.SMALL@|1|The default number of stream to use for a small file
@GRIDFTP.STREAMS.BIG@|3|The default number of stream to use for a big file
@GRIDFTP.FILESIZE.THRESHOLD@|100|The Threshold (in MB) above which a file to transfer is considered "big"
EOF

oldEDG_LOCATION=$EDG_LOCATION
oldEDG_LOCATION_VAR=$EDG_LOCATION_VAR
export EDG_LOCATION=${INSTALL_ROOT}/edg
export EDG_LOCATION_VAR=${INSTALL_ROOT}/edg/var

${INSTALL_ROOT}/edg/sbin/edg-replica-manager-configure \
${INSTALL_ROOT}/edg/etc/edg-replica-manager/edg-replica-manager.conf.values_local >> $YAIM_LOG

export EDG_LOCATION=$oldEDG_LOCATION
export EDG_LOCATION_VAR=$oldEDG_LOCATION_VAR

return 0
}
```

16.7. CONFIG_TAR_USER_ENV

```
function config_tar_user_env () {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if ( ! central_certs ); then
    x509_cert_dir=${INSTALL_ROOT}/etc/grid-security/certificates
else
    x509_cert_dir=${X509_CERT_DIR:-/etc/grid-security/certificates}
fi

if [ ! -d /etc/grid-security/vomsdir -a -d $INSTALL_ROOT/etc/grid-security/vomsdir ]; then
    x509_voms_dir=${INSTALL_ROOT}/etc/grid-security/vomsdir
else
    x509_voms_dir=${X509_VOMS_DIR:-/etc/grid-security/vomsdir}
fi

# If GLOBUS_TCP_PORT_RANGE is unset, give it a good default
GLOBUS_TCP_PORT_RANGE=${GLOBUS_TCP_PORT_RANGE:-"20000 25000"}

####################################
# sh
####################################

cat > $INSTALL_ROOT/etc/profile.d/grid_env.sh <<EOF

# Edit this line to reflect the mountpoint of your middleware
INSTALL_ROOT="$INSTALL_ROOT"
EOF

cat >> $INSTALL_ROOT/etc/profile.d/grid_env.sh <<'EOF'

# Leave the rest...

if [ "${LCG_ENV_SET}" ]; then return 0; fi

# Root directory for EDG software. (mandatory)
# Usual value: /opt/edg
```

```
EDG_LOCATION=${INSTALL_ROOT}/edg
# Directory for machine-specific files.
# Usual value: $EDG_LOCATION/var
EDG_LOCATION_VAR=$EDG_LOCATION/var
# World writable directory for temporary files. (mandatory)
# Usual value: /tmp
EDG_TMP=/tmp
EDG_WL_LOCATION=$EDG_LOCATION


# Usual value: /opt/lcg
LCG_LOCATION=${INSTALL_ROOT}/lcg
# Directory for machine-specific files.
# Usual value: $LCG_LOCATION/var
LCG_LOCATION_VAR=$LCG_LOCATION/var
# World writable directory for temporary files. (mandatory)
# Usual value: /tmp
LCG_TMP=/tmp

GLOBUS_LOCATION=${INSTALL_ROOT}/globus
#GLOBUS_CONFIG=/etc/globus.conf
GLOBUS_CONFIG=/dev/null

GPT_LOCATION=${INSTALL_ROOT}/gpt

GLITE_LOCATION=${INSTALL_ROOT}/glite
GLITE_LOCATION_VAR=${GLITE_LOCATION}/var

for i in ${INSTALL_ROOT}/etc/env.d/*.sh; do
       if [ -x $i ]; then
           . $i
       fi
done

LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${INSTALL_ROOT}/edg/externals/lib

if [ -d ${INSTALL_ROOT}/gcc-3.2.2/lib ]; then
        LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${INSTALL_ROOT}/gcc-3.2.2/lib
fi

RGMA_HOME="${INSTALL_ROOT}/glite"

export EDG_LOCATION EDG_LOCATION_VAR EDG_TMP EDG_WL_LOCATION LCG_LOCATION LCG_LOCATION_VAR LCG_TMP JAVA_HOME GLOBUS

EOF

cat >> $INSTALL_ROOT/etc/profile.d/grid_env.sh <<EOF

GLOBUS_TCP_PORT_RANGE="${GLOBUS_TCP_PORT_RANGE}"
export GLOBUS_TCP_PORT_RANGE

X509_CERT_DIR=$x509_cert_dir
X509_VOMS_DIR=$x509_voms_dir
export X509_CERT_DIR X509_VOMS_DIR
```

```
EOF

####################################
# csh
####################################

cat > $INSTALL_ROOT/etc/profile.d/grid_env.csh <<EOF

# Edit this line to reflect the mountpoint of your middleware
setenv INSTALL_ROOT "$INSTALL_ROOT"
EOF

cat >> $INSTALL_ROOT/etc/profile.d/grid_env.csh <<'EOF'

# Leave the rest...

if ( $?LCG_ENV_SET ) then
    return 0
endif

# Root directory for EDG software. (mandatory)
# Usual value: /opt/edg
setenv EDG_LOCATION ${INSTALL_ROOT}/edg
# Directory for machine-specific files.
# Usual value: $EDG_LOCATION/var
setenv EDG_LOCATION_VAR $EDG_LOCATION/var
# World writable directory for temporary files. (mandatory)
# Usual value: /tmp
setenv EDG_TMP /tmp
setenv EDG_WL_LOCATION $EDG_LOCATION


# Usual value: /opt/lcg
setenv LCG_LOCATION ${INSTALL_ROOT}/lcg
# Directory for machine-specific files.
# Usual value: $LCG_LOCATION/var
setenv LCG_LOCATION_VAR $LCG_LOCATION/var
# World writable directory for temporary files. (mandatory)
# Usual value: /tmp
setenv LCG_TMP /tmp

setenv GLOBUS_LOCATION ${INSTALL_ROOT}/globus
#setenv GLOBUS_CONFIG /etc/globus.conf
setenv GLOBUS_CONFIG /dev/null

setenv GPT_LOCATION ${INSTALL_ROOT}/gpt

setenv GLITE_LOCATION ${INSTALL_ROOT}/glite
setenv GLITE_LOCATION_VAR ${GLITE_LOCATION}/var

foreach i (${INSTALL_ROOT}/etc/env.d/*.csh)
       if ( -x $i ) then
           source $i
       endif
```

```
end

setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:${INSTALL_ROOT}/edg/externals/lib

if ( -d ${INSTALL_ROOT}/gcc-3.2.2/lib ) then
        setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:${INSTALL_ROOT}/gcc-3.2.2/lib
endif

setenv RGMA_HOME "${INSTALL_ROOT}/glite"

EOF

cat >>$INSTALL_ROOT/etc/profile.d/grid_env.csh <<EOF
setenv GLOBUS_TCP_PORT_RANGE "$GLOBUS_TCP_PORT_RANGE"

setenv X509_CERT_DIR $x509_cert_dir
setenv X509_VOMS_DIR $x509_voms_dir
EOF

if [ $UID -eq 0 ]; then
echo "If you want this to be your default middleware installation"
echo "please run the following command"
echo "ln -s $INSTALL_ROOT/etc/profile.d/grid_env.sh $INSTALL_ROOT/etc/profile.d/grid_env.csh /etc/profile.d"
else
echo "IMPORTANT"
echo "you need to make sure that the correct grid_env.*sh file in"
echo "$INSTALL_ROOT/etc/profile.d/"
echo "is sourced when you log in so that your"
echo "environment is set up correctly"
fi


}
```

16.8. CONFIG_LCGENV

```
config_lcgenv() {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if !( echo "${NODE_TYPE_LIST}" | grep TAR > /dev/null ); then
LCG_ENV_LOC=/etc/profile.d
else
LCG_ENV_LOC=${INSTALL_ROOT}/etc/env.d
fi

requires BDII_HOST SITE_NAME CE_HOST

if ( ! echo "${NODE_TYPE_LIST}" | grep -q UI ); then
    requires VOS VO__SW_DIR SE_LIST
fi
```

```
default_se="${SE_LIST%% *}"

if [ "$default_se" ]; then
    for VO in `echo $VOS | tr '[:lower:]' '[:upper:]'`; do
if [ "x`eval echo '$'VO_${VO}_DEFAULT_SE`" = "x" ]; then
    eval VO_${VO}_DEFAULT_SE=$default_se
fi
    done
fi

########## sh ##########
cat << EOF > ${LCG_ENV_LOC}/lcgenv.sh
#!/bin/sh
if test "x\${LCG_ENV_SET+x}" = x; then
export LCG_GFAL_INFOSYS=$BDII_HOST:2170
EOF

if [ "$PX_HOST" ]; then
echo "export MYPROXY_SERVER=$PX_HOST" >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if ( echo "${NODE_TYPE_LIST}" | egrep -q 'WN|VOBOX' ); then
    if [ "$SITE_NAME" ]; then
echo "export SITE_NAME=$SITE_NAME" >> ${LCG_ENV_LOC}/lcgenv.sh
    fi

    if [ "$CE_HOST" ]; then
echo "export SITE_GIIS_URL=$CE_HOST" >> ${LCG_ENV_LOC}/lcgenv.sh
    fi
fi

if [ -d ${INSTALL_ROOT}/d-cache/srm/bin ]; then
    echo export PATH="\${PATH}:${INSTALL_ROOT}/d-cache/srm/bin:${INSTALL_ROOT}/d-cache/dcap/bin" >> ${LCG_ENV_LOC}/
fi

if [ -d ${INSTALL_ROOT}/d-cache/dcap/lib ]; then
    echo export LD_LIBRARY_PATH=\${LD_LIBRARY_PATH}:${INSTALL_ROOT}/d-cache/dcap/lib >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if [ -d ${INSTALL_ROOT}/d-cache/srm ]; then
    echo export SRM_PATH=${INSTALL_ROOT}/d-cache/srm >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if [ "$EDG_WL_SCRATCH" ]; then
    echo "export EDG_WL_SCRATCH=$EDG_WL_SCRATCH"  >> ${LCG_ENV_LOC}/lcgenv.sh
fi

for VO in `echo $VOS | tr '[:lower:]' '[:upper:]'`; do

    default_se=`eval echo '$'VO_${VO}_DEFAULT_SE`
    if [ "$default_se" ]; then
echo "export VO_${VO}_DEFAULT_SE=$default_se" >> ${LCG_ENV_LOC}/lcgenv.sh
    fi
    if ( ! echo "${NODE_TYPE_LIST}" | grep -q UI ); then
sw_dir=`eval echo '$'VO_${VO}_SW_DIR`
if [ "$sw_dir" ]; then
    echo "export VO_${VO}_SW_DIR=$sw_dir" >> ${LCG_ENV_LOC}/lcgenv.sh
fi
    fi

done

if [ "$VOBOX_HOST" ]; then
   requires GSSKLOG
   if  [ "${GSSKLOG}x" == "yesx" ]; then
      requires GSSKLOG_SERVER
      echo "export GSSKLOG_SERVER=$GSSKLOG_SERVER" >> ${LCG_ENV_LOC}/lcgenv.sh
   fi
fi

if [ "${DPM_HOST}" ]; then
    echo "export DPNS_HOST=${DPM_HOST}" >> ${LCG_ENV_LOC}/lcgenv.sh
    echo "export DPM_HOST=${DPM_HOST}" >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if [ "$GLOBUS_TCP_PORT_RANGE" ]; then
    echo "export MYPROXY_TCP_PORT_RANGE=\"${GLOBUS_TCP_PORT_RANGE/ /,}\"" >> ${LCG_ENV_LOC}/lcgenv.sh
fi

if ( echo $NODE_TYPE_LIST | egrep -q UI ); then
    cat << EOF >> ${LCG_ENV_LOC}/lcgenv.sh
if [ "x\$X509_USER_PROXY" = "x" ]; then
    export X509_USER_PROXY=/tmp/x509up_u\$(id -u)
fi
EOF
fi

echo fi >> ${LCG_ENV_LOC}/lcgenv.sh
########## sh ##########


########## csh ##########
cat << EOF > ${LCG_ENV_LOC}/lcgenv.csh
#!/bin/csh
if ( ! \$?LCG_ENV_SET ) then
setenv LCG_GFAL_INFOSYS $BDII_HOST:2170
EOF

if [ "$PX_HOST" ]; then
echo "setenv MYPROXY_SERVER $PX_HOST" >>  ${LCG_ENV_LOC}/lcgenv.csh
fi

if ( echo "${NODE_TYPE_LIST}" | egrep -q 'WN|VOBOX' ); then
    if [ "$SITE_NAME" ]; then
echo "setenv SITE_NAME $SITE_NAME" >>  ${LCG_ENV_LOC}/lcgenv.csh
    fi
    if [ "$CE_HOST" ]; then
echo "setenv SITE_GIIS_URL $CE_HOST" >>  ${LCG_ENV_LOC}/lcgenv.csh
    fi
fi

if [ -d ${INSTALL_ROOT}/d-cache/srm/bin ]; then
    echo setenv PATH "\${PATH}:${INSTALL_ROOT}/d-cache/srm/bin:${INSTALL_ROOT}/d-cache/dcap/bin" >> ${LCG_ENV_LOC}/
fi

if [ -d ${INSTALL_ROOT}/d-cache/dcap/lib ]; then
    echo setenv LD_LIBRARY_PATH \${LD_LIBRARY_PATH}:${INSTALL_ROOT}/d-cache/dcap/lib >> ${LCG_ENV_LOC}/lcgenv.csh
fi

if [ -d ${INSTALL_ROOT}/d-cache/srm ]; then
    echo setenv SRM_PATH ${INSTALL_ROOT}/d-cache/srm >> ${LCG_ENV_LOC}/lcgenv.csh
fi

if [ "$EDG_WL_SCRATCH" ]; then
    echo "setenv EDG_WL_SCRATCH $EDG_WL_SCRATCH"  >> ${LCG_ENV_LOC}/lcgenv.csh
fi

for VO in ‘echo $VOS | tr ’[:lower:]’ ’[:upper:]’‘; do

    default_se=‘eval echo ’$’VO_${VO}_DEFAULT_SE‘
    if [ "$default_se" ]; then
echo "setenv VO_${VO}_DEFAULT_SE $default_se" >> ${LCG_ENV_LOC}/lcgenv.csh
    fi

    if ( ! echo "${NODE_TYPE_LIST}" | grep -q UI ); then
sw_dir=‘eval echo ’$’VO_${VO}_SW_DIR‘
if [ "$sw_dir" ]; then
    echo "setenv VO_${VO}_SW_DIR $sw_dir" >> ${LCG_ENV_LOC}/lcgenv.csh
fi
    fi

done

if [ "$VOBOX_HOST" ]; then
   requires GSSKLOG
   if [ "${GSSKLOG}x" == "yesx" ]; then
   requires GSSKLOG_SERVER
      echo "setenv GSSKLOG_SERVER $GSSKLOG_SERVER" >> ${LCG_ENV_LOC}/lcgenv.csh
   fi
fi


if [ "${DPM_HOST}" ]; then
    echo "setenv DPNS_HOST ${DPM_HOST}" >> ${LCG_ENV_LOC}/lcgenv.csh
    echo "setenv DPM_HOST ${DPM_HOST}" >> ${LCG_ENV_LOC}/lcgenv.csh
fi

if [ "$GLOBUS_TCP_PORT_RANGE" ]; then
    echo "setenv MYPROXY_TCP_PORT_RANGE \"${GLOBUS_TCP_PORT_RANGE/ /,}\"" >> ${LCG_ENV_LOC}/lcgenv.csh
```

```
fi

if ( echo $NODE_TYPE_LIST | egrep -q UI ); then
    cat << EOF >> ${LCG_ENV_LOC}/lcgenv.csh
if ( ! \$?X509_USER_PROXY ) then
   setenv X509_USER_PROXY /tmp/x509up_u\`id -u\`
endif
EOF
fi

echo endif >> ${LCG_ENV_LOC}/lcgenv.csh
########## csh ##########

chmod a+xr ${LCG_ENV_LOC}/lcgenv.csh
chmod a+xr ${LCG_ENV_LOC}/lcgenv.sh

return 0
}
```

16.9. CONFIG_JAVA

```
function config_java () {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

# If JAVA_LOCATION is not set by the admin, take a guess
if [ -z "$JAVA_LOCATION" ]; then
    java=`rpm -qa | grep j2sdk-` || java=`rpm -qa | grep j2re`
    if [ "$java" ]; then
JAVA_LOCATION=`rpm -ql $java | egrep '/bin/java$' | sort | head -1 | sed 's#/bin/java##'`
    fi
fi

if [ ! "$JAVA_LOCATION" -o ! -d "$JAVA_LOCATION" ]; then
   echo "Please check your value for JAVA_LOCATION"
   return 1
fi

if ( echo "${NODE_TYPE_LIST}" | grep TAR > /dev/null ); then

# We're configuring a relocatable distro

    if [ ! -d ${INSTALL_ROOT}/edg/etc/profile.d ]; then
mkdir -p ${INSTALL_ROOT}/edg/etc/profile.d/
    fi

    cat  > $INSTALL_ROOT/edg/etc/profile.d/j2.sh <<EOF

JAVA_HOME=$JAVA_LOCATION
export JAVA_HOME
EOF
```

```
    cat  > $INSTALL_ROOT/edg/etc/profile.d/j2.csh <<EOF

setenv JAVA_HOME $JAVA_LOCATION
EOF

    chmod a+rx $INSTALL_ROOT/edg/etc/profile.d/j2.sh
    chmod a+rx $INSTALL_ROOT/edg/etc/profile.d/j2.csh

    return 0

fi # end of relocatable stuff

# We're root and it's not a relocatable

if [ ! -d /etc/java ]; then
    mkdir /etc/java
fi

echo "export JAVA_HOME=$JAVA_LOCATION" > /etc/java/java.conf
echo "export JAVA_HOME=$JAVA_LOCATION" > /etc/java.conf
chmod +x /etc/java/java.conf

#This hack is here due to SL and the java profile rpms, Laurence Field

if [ ! -d ${INSTALL_ROOT}/edg/etc/profile.d ]; then
    mkdir -p ${INSTALL_ROOT}/edg/etc/profile.d/
fi

cat << EOF > $INSTALL_ROOT/edg/etc/profile.d/j2.sh
if [ -z "\$PATH" ]; then
   export PATH=${JAVA_LOCATION}/bin
else
   export PATH=${JAVA_LOCATION}/bin:\${PATH}
fi
EOF

chmod a+rx $INSTALL_ROOT/edg/etc/profile.d/j2.sh

cat << EOF > $INSTALL_ROOT/edg/etc/profile.d/j2.csh
if ( \$?PATH ) then
    setenv PATH ${JAVA_LOCATION}/bin:\${PATH}
else
    setenv PATH ${JAVA_LOCATION}/bin
endif
EOF

chmod a+rx $INSTALL_ROOT/edg/etc/profile.d/j2.csh

return 0

}
```

16.10. CONFIG_RGMA_CLIENT

```
config_rgma_client(){

requires MON_HOST REG_HOST

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

# NB java stuff now in config_java, which must be run before

export RGMA_HOME=${INSTALL_ROOT}/glite

# in order to use python from userdeps.tgz we need to source the env
if ( echo "${NODE_TYPE_LIST}" | grep TAR > /dev/null ); then
    . $INSTALL_ROOT/etc/profile.d/grid_env.sh
fi

${RGMA_HOME}/share/rgma/scripts/rgma-setup.py --secure=yes --server=${MON_HOST} --registry=${REG_HOST} --schema=${R

cat << EOF > ${INSTALL_ROOT}/edg/etc/profile.d/edg-rgma-env.sh
export RGMA_HOME=${INSTALL_ROOT}/glite
export APEL_HOME=${INSTALL_ROOT}/glite

echo \$PYTHONPATH | grep -q ${INSTALL_ROOT}/glite/lib/python && isthere=1 || isthere=0
if [ \$isthere = 0 ]; then
    if [ -z \$PYTHONPATH ]; then
        export PYTHONPATH=${INSTALL_ROOT}/glite/lib/python
    else
        export PYTHONPATH=\$PYTHONPATH:${INSTALL_ROOT}/glite/lib/python
    fi
fi

echo \$LD_LIBRARY_PATH | grep -q ${INSTALL_ROOT}/glite/lib && isthere=1 || isthere=0
if [ \$isthere = 0 ]; then
    if [ -z \$LD_LIBRARY_PATH ]; then
        export LD_LIBRARY_PATH=${INSTALL_ROOT}/glite/lib
    else
        export LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:${INSTALL_ROOT}/glite/lib
    fi
fi
EOF

chmod a+rx ${INSTALL_ROOT}/edg/etc/profile.d/edg-rgma-env.sh

cat << EOF > ${INSTALL_ROOT}/edg/etc/profile.d/edg-rgma-env.csh
setenv RGMA_HOME ${INSTALL_ROOT}/glite
setenv APEL_HOME ${INSTALL_ROOT}/glite

echo \$PYTHONPATH | grep -q ${INSTALL_ROOT}/glite/lib/python && set isthere=1 || set isthere=0
if ( \$isthere == 0 ) then
    if ( -z \$PYTHONPATH ) then
        setenv PYTHONPATH ${INSTALL_ROOT}/glite/lib/python
    else
        setenv PYTHONPATH \$PYTHONPATH\:${INSTALL_ROOT}/glite/lib/python
    endif
endif

echo \$LD_LIBRARY_PATH | grep -q ${INSTALL_ROOT}/glite/lib && set isthere=1 || set isthere=0
if ( \$isthere == 0 ) then
    if ( -z \$LD_LIBRARY_PATH ) then
        setenv LD_LIBRARY_PATH ${INSTALL_ROOT}/glite/lib
    else
        setenv LD_LIBRARY_PATH \$LD_LIBRARY_PATH\:${INSTALL_ROOT}/glite/lib
    endif
endif
EOF

chmod a+rx  ${INSTALL_ROOT}/edg/etc/profile.d/edg-rgma-env.csh

return 0
}
```

16.11. CONFIG_WORKLOAD_MANAGER_CLIENT

```
function config_workload_manager_client() {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if [ $UID -eq 0 ]; then
    OUTPUT_STORAGE=${OUTPUT_STORAGE:-/tmp/jobOutput}
else
    OUTPUT_STORAGE=${OUTPUT_STORAGE:-${HOME}/jobOutput}
fi

requires PX_HOST RB_HOST VOS

for i in $VOS ; do
if [ ! -d $INSTALL_ROOT/edg/etc/$i ]; then
mkdir -p $INSTALL_ROOT/edg/etc/$i
fi
if [ -f $INSTALL_ROOT/edg/etc/$i/edg_wl_ui.conf ]; then
mv -f $INSTALL_ROOT/edg/etc/$i/edg_wl_ui.conf /tmp/edg_wl_ui.conf.$i.orig
fi
cat <<EOF >$INSTALL_ROOT/edg/etc/$i/edg_wl_ui.conf
[
VirtualOrganisation = "$i";
NSAddresses = "$RB_HOST:7772";
LBAddresses = "$RB_HOST:9000";
## HLR location is optional. Uncomment and fill correctly for
## enabling accounting
#HLRLocation = "fake HLR Location"
## MyProxyServer is optional. Uncomment and fill correctly for
## enabling proxy renewal. This field should be set equal to
## MYPROXY_SERVER environment variable
MyProxyServer = "$PX_HOST"
]
```

```
EOF
done

if [ -f $INSTALL_ROOT/edg/etc/edg_wl_ui_cmd_var.conf ]; then
mv -f $INSTALL_ROOT/edg/etc/edg_wl_ui_cmd_var.conf  /tmp/edg_wl_ui_cmd_var.conf.orig
fi
cat <<EOF >$INSTALL_ROOT/edg/etc/edg_wl_ui_cmd_var.conf
[
rank = - other.GlueCEStateEstimatedResponseTime;
requirements = other.GlueCEStateStatus == "Production";
RetryCount = 3;
ErrorStorage = "/tmp";
OutputStorage = "${OUTPUT_STORAGE}";
ListenerPort = 44000;
ListenerStorage = "/tmp";
LoggingTimeout = 30;
LoggingSyncTimeout = 30;
LoggingDestination = "$RB_HOST:9002";
# Default NS logger level is set to 0 (null)
# max value is 6 (very ugly)
NSLoggerLevel = 0;
DefaultLogInfoLevel = 0;
DefaultStatusLevel = 0;
DefaultVo = "unspecified";
]
EOF

if [ -f $INSTALL_ROOT/edg/etc/edg_wl_ui_gui_var.conf ]; then
mv -f $INSTALL_ROOT/edg/etc/edg_wl_ui_gui_var.conf /tmp/edg_wl_ui_gui_var.conf.orig
fi
cat <<EOF >$INSTALL_ROOT/edg/etc/edg_wl_ui_gui_var.conf
[
JDLEDefaultSchema = "Glue";
Glue = [
rank = - other.GlueCEStateEstimatedResponseTime;
rankMPI = other.GlueCEStateFreeCPUs;
requirements = other.GlueCEStateStatus == "Production"
];
EDG = [
rank = - other.EstimatedTraversalTime;
rankMPI = other.FreeCPUs;
requirements = true
];
RetryCount = 3;
ErrorStorage = "/tmp";
OutputStorage = "$OUTPUT_STORAGE";
ListenerPort = 44000;
ListenerStorage = "/tmp";
LoggingTimeout = 30;
LoggingSyncTimeout = 30;
LoggingDestination = "$RB_HOST:9002";
# Default NS logger level is set to 0 (null)
# max value is 6 (very ugly)
NSLoggerLevel = 0;
DefaultLogInfoLevel = 0;
DefaultStatusLevel = 0;
DefaultVo = "unspecified";
]
EOF

if [ ! -d $OUTPUT_STORAGE ]; then
mkdir $OUTPUT_STORAGE
fi

if [ $UID -eq 0 ]; then
chmod 1777 $OUTPUT_STORAGE
fi

if [ ! -d ${INSTALL_ROOT}/edg/var/etc/profile.d ]; then
mkdir -p ${INSTALL_ROOT}/edg/var/etc/profile.d
fi

for i in edg-wl-ui-env.csh edg-wl-ui-env.sh edg-wl-ui-gui-env.csh edg-wl-ui-gui-env.sh; do
if [ -f ${INSTALL_ROOT}/edg/etc/profile.d/${i} ]; then
cp -f ${INSTALL_ROOT}/edg/etc/profile.d/${i} ${INSTALL_ROOT}/edg/var/etc/profile.d
fi
done

}
```

16.12. CONFIG_VOMSES

```
function config_vomses () {

requires VOS

if [ $UID -eq 0 ]; then
    vomsespath="$INSTALL_ROOT/edg/etc/vomses"
else
    vomsespath="$HOME/.edg/vomses"
fi

if [ "$vomsespath" -a ! -d "$vomsespath" ]; then
    mkdir -p $vomsespath
fi

for vo in $VOS; do
    eval vomses='$'VO_`echo $vo | tr '[:lower:]' '[:upper:]'`_VOMSES
    if [ -z "$vomses" ]; then continue; fi
    split_quoted_variable $vomses | while read line; do
server=`echo $line | awk '{print $2}'`
filename="${vo}-${server}"
echo $line | sed -e 's/ /" "/g' -e 's/^/"/' -e 's/$/"/' > $vomsespath/$filename
chmod 644 $vomsespath/$filename
    done
done
if [ $UID -eq 0 ]; then
    if [ ! -d $INSTALL_ROOT/glite/etc/vomses ]; then
mkdir -p $INSTALL_ROOT/glite/etc/vomses
    fi
    cp -p $vomsespath/* $INSTALL_ROOT/glite/etc/vomses
fi

return 0

}
```

16.13. CONFIG_FTS_CLIENT

```
function config_fts_client () {

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

if [ -z "$FTS_SERVER_URL" ]; then
    return 0
fi

cat > ${INSTALL_ROOT}/glite/etc/services.xml <<EOF
<?xml version="1.0" encoding="UTF-8"?>

<services>

  <service name="EGEEfts">
    <parameters>
      <endpoint>${FTS_SERVER_URL}/services/FileTransfer</endpoint>
      <type>org.glite.FileTransfer</type>
      <version>3.0.0</version>
    </parameters>
  </service>

  <service name="EGEEchannel">
    <parameters>
      <endpoint>${FTS_SERVER_URL}/services/ChannelManagement</endpoint>
      <type>org.glite.ChannelManagement</type>
      <version>3.0.0</version>
    </parameters>
  </service>

</services>
EOF

return 0

}
```