Hosting Environment (Daemon) Services

Generated by Doxygen 1.6.1

Fri Aug 24 03:21:38 2012

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 2

Data Structure Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

Namespace Documentation

4.1 DREService Namespace Reference

Data Structures

- class DREWebService
- class PerlProcessor
- class Task
- class TaskQueue
- class TaskSet

4.1.1 Detailed Description

Implementation of a simple echo service

The reply of the echo service contains the string which was send to it.

Chapter 5

Data Structure Documentation

5.1 ARex::ARexGMConfig Class Reference

The documentation for this class was generated from the following file:

- job.h

5.2 ARex::ARexJob Class Reference

```
#include <job.h>
```

Public Member Functions

- ARexJob (const std::string &id, ARexGMConfig &config, Arc::Logger &logger, bool fast_auth_check=false)
- ARexJob (Arc::XMLNode jsdl, ARexGMConfig &config, const std::string &credentials, const std::string &clientid, Arc::Logger &logger, JobIDGenerator &idgenerator, Arc::XMLNode migration=Arc::XMLNode())
- std::string Failure (void)
- std::string ID (void)
- bool GetDescription (Arc::XMLNode &jsdl)
- bool Cancel (void)
- bool Clean (void)
- bool Resume (void)
- std::string State (void)
- std::string State (bool &job_pending)
- bool Failed (void)
- std::string FailedState (std::string &cause)
- Arc::Time Created (void)
- Arc::Time Modified (void)
- std::string SessionDir (void)
- std::string LogDir (void)
- Arc::FileAccess ∗ CreateFile (const std::string &filename)
- Arc::FileAccess ∗ OpenFile (const std::string &filename, bool for_read, bool for_write)
- int OpenLogFile (const std::string &name)
- Arc::FileAccess ∗ OpenDir (const std::string &dirname)
- std::list< std::string > LogFiles (void)
- bool UpdateCredentials (const std::string &credentials)
- bool ChooseSessionDir (const std::string &jobid, std::string &sessiondir)

Static Public Member Functions

- static int TotalJobs (ARexGMConfig &config, Arc::Logger &logger)
- static std::list< std::string > Jobs (ARexGMConfig &config, Arc::Logger &logger)

5.2.1 Detailed Description

This class represents convenience interface to manage jobs handled by Grid Manager. It works mostly through corresponding classes and functions of Grid Manager.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 ARex::ARexJob::ARexJob (const std::string & *id*, ARexGMConfig & *config*, Arc::Logger & *logger*, bool *fast_auth_check* = `false`)

Create instance which is an interface to existing job

5.2.2.2 ARex::ARexJob::ARexJob (Arc::XMLNode *jsdl*, ARexGMConfig & *config*, const std::string & *credentials*, const std::string & *clientid*, Arc::Logger & *logger*, JobIDGenerator & *idgenerator*, Arc::XMLNode *migration* = `Arc::XMLNode()`)

Create new job with provided JSDL description

5.2.3 Member Function Documentation

5.2.3.1 bool ARex::ARexJob::Cancel (void)

Cancel processing/execution of job

5.2.3.2 bool ARex::ARexJob::ChooseSessionDir (const std::string & *jobid*, std::string & *sessiondir*)

Select a session dir to use for this job

5.2.3.3 bool ARex::ARexJob::Clean (void)

Remove job from local pool

5.2.3.4 Arc::Time ARex::ARexJob::Created (void)

Returns time when job was created.

5.2.3.5 Arc::FileAccess* ARex::ARexJob::CreateFile (const std::string & *filename*)

Creates file in job's session directory and returns handler

5.2.3.6 bool ARex::ARexJob::Failed (void)

Returns true if job has failed

5.2.3.7 std::string ARex::ARexJob::FailedState (std::string & *cause*)

Returns state at which job failed and sets cause to information what caused job failure: "internal" for server initiated and "client" for canceled on client request.

5.2.3.8 std::string ARex::ARexJob::Failure (void) `[inline]`

Returns textual description of failure of last operation

5.2.3.9 bool ARex::ARexJob::GetDescription (Arc::XMLNode & *jsdl*)

Fills provided jsdl with job description

5.2.3.10 std::string ARex::ARexJob::ID (void) `[inline]`

Return ID assigned to job

5.2.3.11 static std::list<std::string> ARex::ARexJob::Jobs (ARexGMConfig & *config*, Arc::Logger & *logger*) `[static]`

Returns list of user's jobs. Fine-grained ACL is ignored.

5.2.3.12 std::string ARex::ARexJob::LogDir (void)

Returns name of virtual log directory

5.2.3.13 std::list<std::string> ARex::ARexJob::LogFiles (void)

Returns list of existing log files

5.2.3.14 Arc::Time ARex::ARexJob::Modified (void)

Returns time when job state was last modified.

5.2.3.15 Arc::FileAccess* ARex::ARexJob::OpenDir (const std::string & *dirname*)

Opens directory inside session directory

5.2.3.16 Arc::FileAccess* ARex::ARexJob::OpenFile (const std::string & *filename*, bool *for_read*, bool *for_write*)

Opens file in job's session directory and returns handler

5.2.3.17 int ARex::ARexJob::OpenLogFile (const std::string & *name*)

Opens log file in control directory

5.2.3.18 bool ARex::ARexJob::Resume (void)

Resume execution of job after error

5.2.3.19 std::string ARex::ARexJob::SessionDir (void)

Returns path to session directory

5.2.3.20 std::string ARex::ARexJob::State (bool & *job_pending*)

Returns current state of job and sets job_pending to true if job is pending due to external limits

5.2.3.21 std::string ARex::ARexJob::State (void)

Returns current state of job

5.2.3.22 static int ARex::ARexJob::TotalJobs (ARexGMConfig & *config*, Arc::Logger & *logger*) `[static]`

Return number of jobs associated with this configuration. TODO: total for all user configurations.

5.2.3.23 bool ARex::ARexJob::UpdateCredentials (const std::string & *credentials*)

Updates job credentials

The documentation for this class was generated from the following file:

- job.h

5.3 ARex::ARexService Class Reference

The documentation for this class was generated from the following file:

- arex.h

5.4 AuthEvaluator Class Reference

The documentation for this class was generated from the following file:

- auth.h

5.5 AuthUser Class Reference

Data Structures

- class **group_t**
- struct **source_t**

The documentation for this class was generated from the following file:

- auth.h

5.6 AuthVO Class Reference

Friends

- class AuthUser

The documentation for this class was generated from the following file:

- auth.h

5.7 CacheConfig Class Reference

```
#include <conf_cache.h>
```

Public Member Functions

- CacheConfig (const GMEnvironment &env, std::string username="")
- CacheConfig ()
- void parseINIConf (std::string username, ConfigSections *cf)
- void setCacheDirs (std::vector< std::string > cache_dirs)

5.7.1 Detailed Description

Reads conf file and provides methods to obtain cache info from it.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 CacheConfig::CacheConfig (const GMEnvironment & *env*, std::string *username* = "")

Create a new CacheConfig instance. Read the config file and fill in private member variables with cache parameters. If different users are defined in the conf file, use the cache parameters for the given username.

5.7.2.2 CacheConfig::CacheConfig () `[inline]`

Empty CacheConfig

5.7.3 Member Function Documentation

5.7.3.1 void CacheConfig::parseINIConf (std::string *username*, ConfigSections * *cf*)

Parsers for the two different conf styles

5.7.3.2 void CacheConfig::setCacheDirs (std::vector< std::string > *cache_dirs*) `[inline]`

To allow for substitutions done during configuration

The documentation for this class was generated from the following file:

- conf_cache.h

5.8 CacheConfigException Class Reference

```
#include <conf_cache.h>
```

5.8.1 Detailed Description

Exception thrown by constructor caused by bad cache params in conf file

The documentation for this class was generated from the following file:

- conf_cache.h

5.9 Cache::CacheService Class Reference

```
#include <CacheService.h>
```

Public Member Functions

- CacheService (Arc::Config ∗cfg, Arc::PluginArgument ∗parg)
- virtual ∼CacheService (void)
- virtual Arc::MCC_Status process (Arc::Message &inmsg, Arc::Message &outmsg)
- bool RegistrationCollector (Arc::XMLNode &doc)
- operator bool ()
- bool operator! ()

Protected Member Functions

- Arc::MCC_Status CacheCheck (Arc::XMLNode in, Arc::XMLNode out, const JobUser &user)
- Arc::MCC_Status CacheLink (Arc::XMLNode in, Arc::XMLNode out, const JobUser &user, const Arc::User &mapped_user)

5.9.1 Detailed Description

CacheService provides functionality for A-REX cache operations that can be performed by remote clients. It currently consists of two operations: CacheCheck - allows querying of the cache for the presence of files. CacheLink - enables a running job to dynamically request cache files to be linked to its working (session) directory. This is especially useful in the case of pilot job workflows where job submission does not follow the usual ARC workflow. In order for input files to be available to jobs, the pilot job can call the cache service to prepare them. If requested files are not present in the cache, they can be downloaded by the cache service if requested, using the A-REX downloader utility.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 Cache::CacheService::CacheService (Arc::Config ∗ *cfg*, Arc::PluginArgument ∗ *parg*)

Make a new CacheService. Reads the configuration and determines the validity of the service.

5.9.2.2 virtual Cache::CacheService::∼CacheService (void) `[virtual]`

Destroy the CacheService

5.9.3 Member Function Documentation

5.9.3.1 Arc::MCC_Status Cache::CacheService::CacheCheck (Arc::XMLNode *in*, Arc::XMLNode *out*, const JobUser & *user*) `[protected]`

Check whether the URLs supplied in the input are present in any cache. Returns in the out message for each file true or false, and if true, the size of the file on cache disk.

Parameters:

user A-REX user configuration for the mapped user

5.9.3.2 Arc::MCC_Status Cache::CacheService::CacheLink (Arc::XMLNode *in*, Arc::XMLNode *out*, const JobUser & *user*, const Arc::User & *mapped_user*) `[protected]`

This method is used to link cache files to the session dir. A list of URLs is supplied and if they are present in the cache and the user calling the service has permission to access them, then they are linked to the given session directory. If the user requests that missing files be staged, then a downloader process is launched to obtain them.

Parameters:

user A-REX user configuration for the mapped user

mapped_user The local user to which the client DN was mapped

5.9.3.3 Cache::CacheService::operator bool (void) `[inline]`

Returns true if the CacheService is valid.

5.9.3.4 bool Cache::CacheService::operator! (void) `[inline]`

Returns true if the CacheService is not valid.

5.9.3.5 virtual Arc::MCC_Status Cache::CacheService::process (Arc::Message & *inmsg*, Arc::Message & *outmsg*) `[virtual]`

Main method called by HED when CacheService is invoked. Directs call to appropriate CacheService method.

5.9.3.6 bool Cache::CacheService::RegistrationCollector (Arc::XMLNode & *doc*)

Supplies information on the service for use in the information system.

The documentation for this class was generated from the following file:

- CacheService.h

5.10 ArcSec::Charon Class Reference

```
#include <charon.h>
```

Data Structures

- class PolicyLocation

5.10.1 Detailed Description

A Service which includes the ArcPDP functionality; it can be deployed as an independent service to provide request evaluation functionality for the other remote services

The documentation for this class was generated from the following file:

- charon.h

5.11 CommFIFO Class Reference

Data Structures

- class **elem_t**

The documentation for this class was generated from the following file:

- commfifo.h

5.12 gridftpd::ConfigSections Class Reference

The documentation for this class was generated from the following file:

- gridftpd/conf/conf_sections.h

5.13 ConfigSections Class Reference

The documentation for this class was generated from the following file:

- a-rex/grid-manager/conf/conf_sections.h

5.14 ContinuationPlugins Class Reference

Data Structures

- class **command_t**
- class result_t

The documentation for this class was generated from the following file:

- plugins.h

5.15 ARex::CountedResource Class Reference

The documentation for this class was generated from the following file:

- arex.h

5.16 gridftpd::Daemon Class Reference

The documentation for this class was generated from the following file:

- daemon.h

5.17 DataStaging::DataDeliveryService Class Reference

Service for the Delivery layer of data staging.

```
#include <DataDeliveryService.h>
```

Public Member Functions

- DataDeliveryService (Arc::Config ∗cfg, Arc::PluginArgument ∗parg)
- virtual ∼DataDeliveryService ()
- virtual Arc::MCC_Status process (Arc::Message &inmsg, Arc::Message &outmsg)
- virtual void receiveDTR (DTR_ptr dtr)
- bool RegistrationCollector (Arc::XMLNode &doc)

5.17.1 Detailed Description

Service for the Delivery layer of data staging. This service starts and controls data transfers. It assumes that the files in any request submitted are ready for immediate transfer and so do not need to be resolved or prepared in any way.

It implements DTRCallback to get callbacks when a DTR has finished transfer.

Status codes in results returned:

- OK - successful submission/cancellation
- TRANSFERRING - transfer still ongoing
- TRANSFERRED - transfer finished successfully
- TRANSFER_ERROR - transfer failed
- SERVICE_ERROR - something went wrong in the service itself

An internal list of active transfers is held in memory. After the first query of a finished transfer (successful or not) the DTR is moved to an archived list where only summary information is kept about the transfer (DTR ID, state and short error description). The DTR object is then deleted. This archived list is also kept in memory. In case a transfer is never queried, a separate thread moves any transfers which completed more than one hour ago to the archived list.

The documentation for this class was generated from the following file:

- DataDeliveryService.h

5.18 ARex::DelegationStore Class Reference

Data Structures

- class **Consumer**

The documentation for this class was generated from the following file:

- DelegationStore.h

5.19 ARex::DelegationStores Class Reference

The documentation for this class was generated from the following file:

- DelegationStores.h

5.20 DirectAccess::diraccess_t Struct Reference

The documentation for this struct was generated from the following file:

- fileplugin.h

5.21 DirectAccess Class Reference

Data Structures

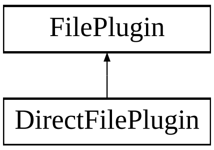
- struct diraccess_t

The documentation for this class was generated from the following file:

- fileplugin.h

5.22 DirectFilePlugin Class Reference

Inheritance diagram for DirectFilePlugin::



The documentation for this class was generated from the following file:

- fileplugin.h

5.23 DirEntry Class Reference

The documentation for this class was generated from the following file:

- fileroot.h

5.24 DREService::DREWebService Class Reference

Public Member Functions

- DREWebService (Arc::Config ∗cfg)
- virtual ∼DREWebService (void)
- virtual Arc::MCC_Status process (Arc::Message &inmsg, Arc::Message &outmsg)

Protected Member Functions

- Arc::MCC_Status makeFault (Arc::Message &outmsg, const std::string &reason)

Protected Attributes

- Arc::NS ns_

Static Protected Attributes

- static Arc::Logger logger

5.24.1 Constructor & Destructor Documentation

5.24.1.1 DREService::DREWebService::DREWebService (Arc::Config ∗ *cfg*)

Constructor which is capable to extract prefix and suffix for the echo service.

5.24.1.2 virtual DREService::DREWebService::∼DREWebService (void) `[virtual]`

Destructor.

5.24.2 Member Function Documentation

5.24.2.1 Arc::MCC_Status DREService::DREWebService::makeFault (Arc::Message & *outmsg*, const std::string & *reason*) `[protected]`

Method to return an error. Creates a fault message and returns a status.

Parameters:

outmsg outgoing message

Returns:

Status of the result achieved

5.24.2.2 virtual Arc::MCC_Status DREService::DREWebService::process (Arc::Message & *inmsg*, Arc::Message & *outmsg*) `[virtual]`

Implementation of the virtual method defined in MCCInterface (to be found in MCC.h).

Parameters:

inmsg incoming message

inmsg outgoing message

Returns:

Status of the result achieved

5.24.3 Field Documentation

5.24.3.1 Arc::Logger DREService::DREWebService::logger `[static, protected]`

Arc-intern logger. Generates output into the file specified in the arched configuration file used to invoke arched services.

5.24.3.2 Arc::NS DREService::DREWebService::ns_ `[protected]`

Class which specifies a XML namespace i.e. "echo". Needed to extract the content out of the incoming message

The documentation for this class was generated from the following file:

- dREWebService.h

5.25 DTRGenerator Class Reference

`#include <dtr_generator.h>`

Public Member Functions

- DTRGenerator (const JobUsers &users, void(∗kicker_func)(void ∗)=NULL, void ∗kicker_arg=NULL)
- ∼DTRGenerator ()
- virtual void receiveDTR (DataStaging::DTR_ptr dtr)
- void receiveJob (const JobDescription &job)
- void cancelJob (const JobDescription &job)
- bool queryJobFinished (JobDescription &job)
- bool hasJob (const JobDescription &job)
- void removeJob (const JobDescription &job)
- int checkUploadedFiles (JobDescription &job)

5.25.1 Detailed Description

A-REX implementation of DTR Generator. Note that neither Janitor nor job migration functionality present in the down/uploaders has been implemented here.

5.25.2 Constructor & Destructor Documentation

5.25.2.1 DTRGenerator::DTRGenerator (const JobUsers & *users*, void(∗)(void ∗) *kicker_func* = `NULL`, void ∗ *kicker_arg* = `NULL`)

Start up Generator.

Parameters:

user JobUsers for this Generator.

kicker_func Function to call on completion of all DTRs for a job

kicker_arg Argument to kicker function

5.25.2.2 DTRGenerator::∼DTRGenerator ()

Stop Generator

5.25.3 Member Function Documentation

5.25.3.1 void DTRGenerator::cancelJob (const JobDescription & *job*)

This method is used by A-REX to cancel on-going DTRs. A cancel request is made for each DTR in the job and the method returns. The Scheduler asychronously deals with cancelling the DTRs.

Parameters:

job The job which is being cancelled

5.25.3.2 int DTRGenerator::checkUploadedFiles (JobDescription & *job*)

Utility method to check that all files the user was supposed to upload with the job are ready.

Parameters:

job Job description, failures will be reported directly in this object.

Returns:

0 if file exists, 1 if it is not a proper file or other error, 2 if the file not there yet

5.25.3.3 bool DTRGenerator::hasJob (const JobDescription & *job*)

Query whether the Generator has a record of this job.

Parameters:

job Job to query.

Returns:

True if the job is active or finished.

5.25.3.4 bool DTRGenerator::queryJobFinished (JobDescription & *job*)

Query status of DTRs in job. If all DTRs are finished, returns true, otherwise returns false. If true is returned, the JobDescription should be checked for whether the staging was successful or not by checking GetFailure().

Parameters:

job Description of job to query. Can be modified to add a failure reason.

Returns:

True if all DTRs in the job are finished, false otherwise.

5.25.3.5 virtual void DTRGenerator::receiveDTR (DataStaging::DTR_ptr *dtr*) `[virtual]`

Callback called when DTR is finished. This DTR is marked done in the DTR list and if all DTRs for the job have completed, the job is marked as done.

Parameters:

dtr DTR object sent back from the Scheduler

5.25.3.6 void DTRGenerator::receiveJob (const JobDescription & *job*)

A-REX sends data transfer requests to the data staging system through this method. It reads the job.id.input/output files, forms DTRs and sends them to the Scheduler.

Parameters:

job Job description object.

5.25.3.7 void DTRGenerator::removeJob (const JobDescription & *job*)

Remove the job from the Generator. Only finished jobs will be removed, and a warning will be logged if the job still has active DTRs. This method should be called after A-REX has finished PREPARING or FINISHING.

Parameters:

job The job to remove.

The documentation for this class was generated from the following file:

- dtr_generator.h

5.26 DTRInfo Class Reference

```
#include <dtr_generator.h>
```

Public Member Functions

- DTRInfo (const JobUsers &users)

5.26.1 Detailed Description

DTRInfo passes state information from data staging to A-REX via the defined callback, called when the DTR passes to the certain processes. It could for example write to files in the control directory, and this information can be picked up and published by the info system.

5.26.2 Constructor & Destructor Documentation

5.26.2.1 DTRInfo::DTRInfo (const JobUsers & *users*)

JobUsers is needed to find the correct control dir

The documentation for this class was generated from the following file:

- dtr_generator.h

5.27 Entry Class Reference

The documentation for this class was generated from the following file:

- Entry.h

5.28 Exec Class Reference

The documentation for this class was generated from the following file:

- info_types.h

5.29 ARex::FileChunks Class Reference

Representation of delivered file chunks.

```
#include <FileChunks.h>
```

Public Member Functions

- std::string Path (void)
- void Size (off_t size)
- off_t Size (void)
- void Add (off_t start, off_t csize)
- bool Complete (void)
- void Print (void)
- void Release (void)
- void Remove (void)

5.29.1 Detailed Description

Representation of delivered file chunks.

5.29.2 Member Function Documentation

5.29.2.1 void ARex::FileChunks::Release (void)

Release reference obtained through FileChunksList::Get() method. This operation may lead to destruction of FileChunk instance hence previously obtained refrence mus tnot be used.

5.29.2.2 void ARex::FileChunks::Remove (void)

Relases reference obtained through Get() method and destroys its instance. Normally this method to be called instead of Release() after whole file is delivered in order to free resources associated with FileChunks instance.

The documentation for this class was generated from the following file:

- FileChunks.h

5.30 ARex::FileChunksList Class Reference

Container for FileChunks instances.

```
#include <FileChunks.h>
```

Public Member Functions

- FileChunks & Get (std::string path)
- void Timeout (int t)

5.30.1 Detailed Description

Container for FileChunks instances.

5.30.2 Member Function Documentation

5.30.2.1 FileChunks& ARex::FileChunksList::Get (std::string *path*)

Returns previously created FileChunks object with associated path. If such instance does not exist new one is created. Obtained reference may be used for other operations. Obtained reference must be Release()ed after it is not longer needed.

The documentation for this class was generated from the following file:

- FileChunks.h

5.31 ARex::FileChunksRef Class Reference

The documentation for this class was generated from the following file:

- FileChunks.h

5.32 FileData Class Reference

The documentation for this class was generated from the following file:

- info_types.h

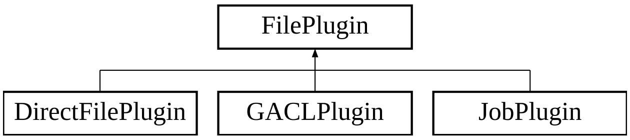
5.33 FileNode Class Reference

The documentation for this class was generated from the following file:

- fileroot.h

5.34 FilePlugin Class Reference

Inheritance diagram for FilePlugin::



The documentation for this class was generated from the following file:

- fileroot.h

5.35 ARex::FileRecord Class Reference

Data Structures

- class Iterator

The documentation for this class was generated from the following file:

- FileRecord.h

5.36 FileRoot Class Reference

Data Structures

- class ServerParams

Friends

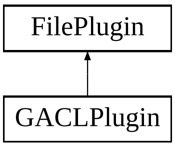
- class GridFTP_Commands

The documentation for this class was generated from the following file:

- fileroot.h

5.37 GACLPlugin Class Reference

Inheritance diagram for GACLPlugin::



The documentation for this class was generated from the following file:

- gaclplugin.h

5.38 gm_dirs_ Struct Reference

The documentation for this struct was generated from the following file:

- jobplugin.h

5.39 GMEnvironment Class Reference

Public Member Functions

- std::string nordugrid_loc (void) const
- std::string nordugrid_data_loc (void) const
- std::string nordugrid_libexec_loc (void) const
- std::string nordugrid_config_loc (void) const
- std::string support_mail_address (void) const

5.39.1 Member Function Documentation

5.39.1.1 std::string GMEnvironment::nordugrid_config_loc (void) const

ARC configuration file /etc/arc.conf $ARC_LOCATION/etc/arc.conf

5.39.1.2 std::string GMEnvironment::support_mail_address (void) const

Email address of person responsible for this ARC installation grid.manager, it can also be set from configuration file

The documentation for this class was generated from the following file:

- a-rex/grid-manager/conf/environment.h

5.40 gridftpd::GMEnvironment Class Reference

Public Member Functions

- std::string nordugrid_loc (void) const
- std::string nordugrid_libexec_loc (void) const
- std::string nordugrid_config_loc (void) const
- std::string support_mail_address (void) const

5.40.1 Member Function Documentation

5.40.1.1 std::string gridftpd::GMEnvironment::nordugrid_config_loc (void) const

ARC configuration file /etc/arc.conf $ARC_LOCATION/etc/arc.conf

5.40.1.2 std::string gridftpd::GMEnvironment::support_mail_address (void) const

Email address of person responsible for this ARC installation grid.manager, it can also be set from configuration file

The documentation for this class was generated from the following file:

- gridftpd/conf/environment.h

5.41 GridFTP_Commands Class Reference

Data Structures

- class **close_semaphor_t**
- struct **data_buffer_t**

Friends

- class GridFTP_Commands_timeout

The documentation for this class was generated from the following file:

- commands.h

5.42 GridFTP_Commands_timeout Class Reference

The documentation for this class was generated from the following file:

- commands.h

5.43 ARex::GridManager Class Reference

The documentation for this class was generated from the following file:

- grid_manager.h

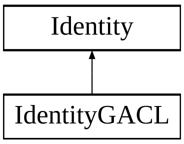
5.44 Hopi::Hopi Class Reference

The documentation for this class was generated from the following file:

- hopi.h

5.45 Identity Class Reference

Inheritance diagram for Identity::



Data Structures

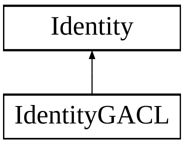
- class Item

The documentation for this class was generated from the following file:

- identity.h

5.46 IdentityGACL Class Reference

Inheritance diagram for IdentityGACL::

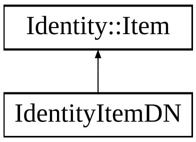


The documentation for this class was generated from the following file:

- identity_gacl.h

5.47 IdentityItemDN Class Reference

Inheritance diagram for IdentityItemDN::

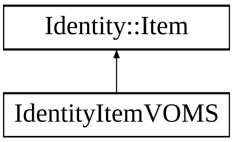


The documentation for this class was generated from the following file:

- identity_dn.h

5.48 IdentityItemVOMS Class Reference

Inheritance diagram for IdentityItemVOMS::



The documentation for this class was generated from the following file:

- identity_voms.h

5.49 Index Class Reference

The documentation for this class was generated from the following file:

- Index.h

5.50 ISIS::ISIService Class Reference

The documentation for this class was generated from the following file:

- isis.h

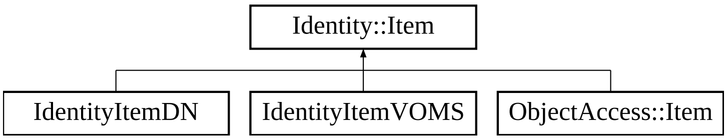
5.51 ISIS::ISISSecAttr Class Reference

The documentation for this class was generated from the following file:

- security.h

5.52 Identity::Item Class Reference

Inheritance diagram for Identity::Item::

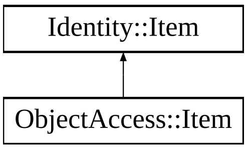


The documentation for this class was generated from the following file:

- identity.h

5.53 ObjectAccess::Item Class Reference

Inheritance diagram for ObjectAccess::Item::



The documentation for this class was generated from the following file:

- object_access.h

5.54 ARex::FileRecord::Iterator Class Reference

The documentation for this class was generated from the following file:

- FileRecord.h

5.55 Janitor Class Reference

Class to communicate with Janitor - Dynmaic Runtime Environment handler.

```
#include <janitor.h>
```

Public Member Functions

- Janitor (const std::string &id, const std::string &cdir, const GMEnvironment &env)
- bool enabled ()
- operator bool (void)
- bool operator! (void)
- bool deploy (void)
- bool remove (void)
- bool wait (int timeout)
- Result result (void)

5.55.1 Detailed Description

Class to communicate with Janitor - Dynmaic Runtime Environment handler.

5.55.2 Constructor & Destructor Documentation

5.55.2.1 Janitor::Janitor (const std::string & *id*, const std::string & *cdir*, const GMEnvironment & *env*)

Creates instance representing job entry in Janitor database. Takes id for job identifier and cdir for the control directory of A-Rex. constructor does not register job in the Janitor. It only associates job with this instance.

5.55.3 Member Function Documentation

5.55.3.1 bool Janitor::deploy (void)

Registers associated job with Janitor and deploys dynamic RTEs. This operation is asynchronous. Returned true means Janitor will be contacted and deployemnt will start soon. For obtaining result of operation see methods wait() and result(). During this operation janitor utility is called with command register and optionally deploy.

5.55.3.2 bool Janitor::remove (void)

Removes job from those handled by Janitor and releases associated RTEs. This operation is asynchronous. Returned true means Janitor will be contacted and removal will start soon. For obtaining result of operation see methods wait() and result(). During this operation janitor utility is called with command remove.

5.55.3.3 Result Janitor::result (void)

Returns true if operation initiated by deploy() or remove() succeeded. It should be called after wait() returned true.

5.55.3.4 bool Janitor::wait (int *timeout*)

Wait till operation initiated by deploy() or remove() finished. This operation returns true if operation finished or false if timeout seconds passed. It may be called repeatedly and even after it previously returned true. If no operation is running it returns true immeaditely.

The documentation for this class was generated from the following file:

- janitor.h

5.56 job_state_rec_t Struct Reference

The documentation for this struct was generated from the following file:

- grid-manager/jobs/job.h

5.57 JobDescription Class Reference

Friends

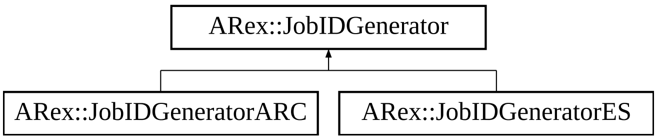
- class JobsList

The documentation for this class was generated from the following file:

- grid-manager/jobs/job.h

5.58 ARex::JobIDGenerator Class Reference

Inheritance diagram for ARex::JobIDGenerator::

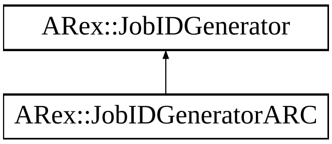


The documentation for this class was generated from the following file:

- tools.h

5.59 ARex::JobIDGeneratorARC Class Reference

Inheritance diagram for ARex::JobIDGeneratorARC::

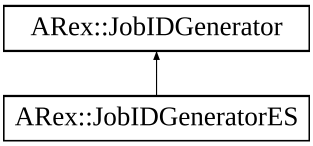


The documentation for this class was generated from the following file:

- tools.h

5.60 ARex::JobIDGeneratorES Class Reference

Inheritance diagram for ARex::JobIDGeneratorES::



The documentation for this class was generated from the following file:

- tools.h

5.61 JobLocalDescription Class Reference

The documentation for this class was generated from the following file:

- info_types.h

5.62 JobLog Class Reference

```
#include <job_log.h>
```

5.62.1 Detailed Description

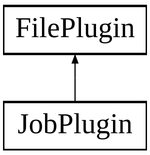
Put short information into log when every job starts/finishes. And store more detailed information for Reporter.

The documentation for this class was generated from the following file:

- job_log.h

5.63 JobPlugin Class Reference

Inheritance diagram for JobPlugin::



The documentation for this class was generated from the following file:

- jobplugin.h

5.64 JobsList Class Reference

The documentation for this class was generated from the following file:

- states.h

5.65 JobsListConfig Class Reference

```
#include <job_config.h>
```

Friends

- class JobsList

5.65.1 Detailed Description

Class to represent information read from configuration.

The documentation for this class was generated from the following file:

- job_config.h

5.66 JobUser Class Reference

The documentation for this class was generated from the following file:

- users.h

5.67 JobUserHelper Class Reference

The documentation for this class was generated from the following file:

- users.h

5.68 JobUsers Class Reference

The documentation for this class was generated from the following file:

- users.h

5.69 gridftpd::LdapQuery Class Reference

```
#include <ldapquery.h>
```

Public Types

- enum Scope

Public Member Functions

- LdapQuery (const std::string &ldaphost, int ldapport, bool anonymous=true, const std::string &usersn="", int timeout=20)
- ∼LdapQuery ()
- void Query (const std::string &base, const std::string &filter="(objectclass=∗)", const std::vector< std::string > &attributes=std::vector< std::string >(), Scope scope=subtree) throw (LdapQueryError)
- void Result (ldap_callback callback, void ∗ref) throw (LdapQueryError)
- std::string Host ()

5.69.1 Detailed Description

LdapQuery class; querying of LDAP servers.

5.69.2 Member Enumeration Documentation

5.69.2.1 enum gridftpd::LdapQuery::Scope

Scope for a LDAP queries. Use when querying.

5.69.3 Constructor & Destructor Documentation

5.69.3.1 gridftpd::LdapQuery::LdapQuery (const std::string & *ldaphost*, int *ldapport*, bool *anonymous* = `true`, const std::string & *usersn* = `""`, int *timeout* = `20`)

Constructs a new LdapQuery object and sets connection options. The connection is first established when calling Query.

5.69.3.2 gridftpd::LdapQuery::∼LdapQuery ()

Destructor. Will disconnect from the ldapserver if stll connected.

5.69.4 Member Function Documentation

5.69.4.1 std::string gridftpd::LdapQuery::Host ()

Returns the hostname of the ldap-server.

5.69.4.2 void gridftpd::LdapQuery::Query (const std::string & *base*, const std::string & *filter* = "(objectclass=∗)", const std::vector< std::string > & *attributes* = std::vector< std::string >(), Scope *scope* = subtree) throw (LdapQueryError)

Queries the ldap server.

5.69.4.3 void gridftpd::LdapQuery::Result (ldap_callback *callback*, void ∗ *ref*) throw (LdapQueryError)

Retrieves the result of the query from the ldap-server.

The documentation for this class was generated from the following file:

- ldapquery.h

5.70 gridftpd::LdapQueryError Class Reference

```
#include <ldapquery.h>
```

Public Member Functions

- LdapQueryError (std::string message)

5.70.1 Detailed Description

LdapQuery exception. Gets thrown whan an error occurs in a query.

5.70.2 Constructor & Destructor Documentation

5.70.2.1 gridftpd::LdapQueryError::LdapQueryError (std::string *message*) `[inline]`

Standard exception class constructor.

The documentation for this class was generated from the following file:

- ldapquery.h

5.71 RunPlugin::lib_plugin_t Union Reference

The documentation for this union was generated from the following file:

- a-rex/grid-manager/run/run_plugin.h

5.72 gridftpd::RunPlugin::lib_plugin_t Union Reference

The documentation for this union was generated from the following file:

- gridftpd/run/run_plugin.h

5.73 LRMSResult Class Reference

The documentation for this class was generated from the following file:

- info_types.h

5.74 ISIS::Neighbor_Container Class Reference

The documentation for this class was generated from the following file:

- isis.h

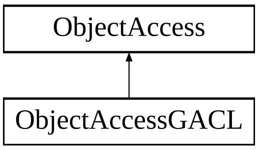
5.75 numvalue_for_shell Class Reference

The documentation for this class was generated from the following file:

- job_desc.h

5.76 ObjectAccess Class Reference

Inheritance diagram for ObjectAccess::



Data Structures

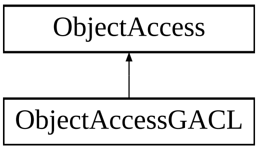
- class Item

The documentation for this class was generated from the following file:

- object_access.h

5.77 ObjectAccessGACL Class Reference

Inheritance diagram for ObjectAccessGACL::



The documentation for this class was generated from the following file:

- object_access_gacl.h

5.78 ARex::OptimizedInformationContainer Class Reference

The documentation for this class was generated from the following file:

- arex.h

5.79 gridftpd::ParallelLdapQueries Class Reference

```
#include <ldapquery.h>
```

5.79.1 Detailed Description

General method to perform parallel ldap-queries to a set of clusters

The documentation for this class was generated from the following file:

- ldapquery.h

5.80 ARex::PayloadBigFile Class Reference

Public Member Functions

- PayloadBigFile (const char ∗filename, Size_t start, Size_t end)
- virtual ∼PayloadBigFile (void)

5.80.1 Constructor & Destructor Documentation

5.80.1.1 ARex::PayloadBigFile::PayloadBigFile (const char ∗ *filename*, Size_t *start*, Size_t *end*)

Creates object associated with file for reading from it

5.80.1.2 virtual ARex::PayloadBigFile::∼PayloadBigFile (void) `[virtual]`

Creates object associated with file for writing into it. Use size=-1 for undefined size.

The documentation for this class was generated from the following file:

- a-rex/PayloadFile.h

5.81 Hopi::PayloadBigFile Class Reference

Public Member Functions

- PayloadBigFile (const char ∗filename, Size_t start, Size_t end)
- virtual ∼PayloadBigFile (void)

5.81.1 Constructor & Destructor Documentation

5.81.1.1 Hopi::PayloadBigFile::PayloadBigFile (const char ∗ *filename*, Size_t *start*, Size_t *end*)

Creates object associated with file for reading from it

5.81.1.2 virtual Hopi::PayloadBigFile::∼PayloadBigFile (void) `[virtual]`

Creates object associated with file for writing into it. Use size=-1 for undefined size.

The documentation for this class was generated from the following file:

- hopi/PayloadFile.h

5.82 ARex::PayloadFAFile Class Reference

Public Member Functions

- PayloadFAFile (Arc::FileAccess ∗h, Size_t start, Size_t end)

5.82.1 Constructor & Destructor Documentation

5.82.1.1 ARex::PayloadFAFile::PayloadFAFile (Arc::FileAccess ∗ *h*, Size_t *start*, Size_t *end*)

Creates object associated with file for reading from it

The documentation for this class was generated from the following file:

- a-rex/PayloadFile.h

5.83 Hopi::PayloadFile Class Reference

```
#include <PayloadFile.h>
```

Public Member Functions

- PayloadFile (const char ∗filename, Size_t start, Size_t end)
- virtual ∼PayloadFile (void)

5.83.1 Detailed Description

Implementation of PayloadRawInterface which provides access to ordinary file. Currently only read-only mode is supported.

5.83.2 Constructor & Destructor Documentation

5.83.2.1 Hopi::PayloadFile::PayloadFile (const char ∗ *filename*, Size_t *start*, Size_t *end*)

Creates object associated with file for reading from it. Use end=-1 for full size.

5.83.2.2 virtual Hopi::PayloadFile::∼PayloadFile (void) `[virtual]`

Creates object associated with file for writing into it. Use size=-1 for undefined size.

The documentation for this class was generated from the following file:

- hopi/PayloadFile.h

5.84 ARex::PayloadFile Class Reference

```
#include <PayloadFile.h>
```

Public Member Functions

- PayloadFile (const char ∗filename, Size_t start, Size_t end)
- virtual ∼PayloadFile (void)

5.84.1 Detailed Description

Implementation of PayloadRawInterface which provides access to ordinary file. Currently only read-only mode is supported.

5.84.2 Constructor & Destructor Documentation

5.84.2.1 ARex::PayloadFile::PayloadFile (const char ∗ *filename*, Size_t *start*, Size_t *end*)

Creates object associated with file for reading from it. Use end=-1 for full size.

5.84.2.2 virtual ARex::PayloadFile::∼PayloadFile (void) `[virtual]`

Creates object associated with file for writing into it. Use size=-1 for undefined size.

The documentation for this class was generated from the following file:

- a-rex/PayloadFile.h

5.85 DREService::PerlProcessor Class Reference

Data Structures

- struct **ThreadInterface**

Public Member Functions

- PerlProcessor (int threadNumber, TaskQueue ∗pTaskQueue, TaskSet ∗pTaskSet)
- virtual ∼PerlProcessor (void)

5.85.1 Constructor & Destructor Documentation

5.85.1.1 DREService::PerlProcessor::PerlProcessor (int *threadNumber*, TaskQueue ∗ *pTaskQueue*, TaskSet ∗ *pTaskSet*)

Constructor which is capable to extract prefix and suffix for the echo service.

5.85.1.2 virtual DREService::PerlProcessor::∼PerlProcessor (void) `[virtual]`

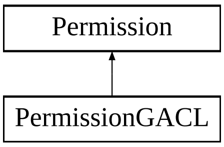
Destructor.

The documentation for this class was generated from the following file:

- PerlProcessor.h

5.86 Permission Class Reference

Inheritance diagram for Permission::

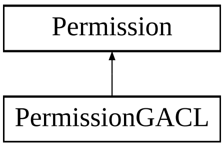


The documentation for this class was generated from the following file:

- permission.h

5.87 PermissionGACL Class Reference

Inheritance diagram for PermissionGACL::



The documentation for this class was generated from the following file:

- permission_gacl.h

5.88 Policy Class Reference

The documentation for this class was generated from the following file:

- Policy.h

5.89 ArcSec::Charon::PolicyLocation Class Reference

The documentation for this class was generated from the following file:

- charon.h

5.90 ContinuationPlugins::result_t Class Reference

The documentation for this class was generated from the following file:

- plugins.h

5.91 RunParallel Class Reference

The documentation for this class was generated from the following file:

- run_parallel.h

5.92 gridftpd::RunPlugin Class Reference

Data Structures

- union lib_plugin_t

The documentation for this class was generated from the following file:

- gridftpd/run/run_plugin.h

5.93 RunPlugin Class Reference

Data Structures

- union lib_plugin_t

The documentation for this class was generated from the following file:

- a-rex/grid-manager/run/run_plugin.h

5.94 RunPlugins Class Reference

The documentation for this class was generated from the following file:

- a-rex/grid-manager/run/run_plugin.h

5.95 RunRedirected Class Reference

The documentation for this class was generated from the following file:

- run_redirected.h

5.96 Server Class Reference

The documentation for this class was generated from the following file:

- Server.h

5.97 FileRoot::ServerParams Class Reference

The documentation for this class was generated from the following file:

- fileroot.h

5.98 ArcSec::Service_AA Class Reference

```
#include <aaservice.h>
```

5.98.1 Detailed Description

A Service which includes the AttributeAuthority functionality; it accepts the <samlp:AttributeQuery> which includes the <Subject> of the principal from the request and <Attribute> which the request would get; it access some local attribute database and returns <samlp:Assertion> which includes the <Attribute>

The documentation for this class was generated from the following file:

- aaservice.h

5.99 Arc::Service_JavaWrapper Class Reference

Public Member Functions

- virtual Arc::MCC_Status process (Arc::Message &, Arc::Message &)

5.99.1 Member Function Documentation

5.99.1.1 virtual Arc::MCC_Status Arc::Service_JavaWrapper::process (Arc::Message &, Arc::Message &) `[virtual]`

Service request processing routine

The documentation for this class was generated from the following file:

- javawrapper.h

5.100 Arc::Service_PythonWrapper Class Reference

Public Member Functions

- virtual Arc::MCC_Status process (Arc::Message &, Arc::Message &)

5.100.1 Member Function Documentation

5.100.1.1 virtual Arc::MCC_Status Arc::Service_PythonWrapper::process (Arc::Message &, Arc::Message &) `[virtual]`

Service request processing routine

The documentation for this class was generated from the following file:

- pythonwrapper.h

5.101 ArcSec::Service_SLCS Class Reference

```
#include <slcs.h>
```

5.101.1 Detailed Description

A Service which signs the short-lived certificate; it accepts the certificate signing request (CSR) from from client side through soap, signs a short-lived certificate and sends back through soap. This service is supposed to be deployed together with the SPService and saml2sso.serviceprovider handler, in order to sign certificate based on the authentication result from saml2sso profile. Also the saml attribute (inside the saml assertion from saml2sso profile) will be put into the signed short-lived certificate. By deploying this service together with SPService and saml2sso.serviceprovider handler, we can get the convertion from username/password ------> x509 certificate.

The documentation for this class was generated from the following file:

- slcs.h

5.102 SPService::Service_SP Class Reference

```
#include <SPService.h>
```

Public Member Functions

- Service_SP (Arc::Config ∗cfg)
- virtual Arc::MCC_Status process (Arc::Message &, Arc::Message &)

5.102.1 Detailed Description

This is service which accepts HTTP request from user agent (web browser) in the client side and processes the functionality of Service Provider in SAML2 SSO profile --- composing <AuthnRequest> Note: the IdP name is provided by the user agent directely when it gives a request, instead of the WRYF(where are you from) or Discovery Service in other implementation

5.102.2 Constructor & Destructor Documentation

5.102.2.1 SPService::Service_SP::Service_SP (Arc::Config ∗ *cfg*)

Constructor

5.102.3 Member Function Documentation

5.102.3.1 virtual Arc::MCC_Status SPService::Service_SP::process (Arc::Message &, Arc::Message &) `[virtual]`

Service request processing routine

The documentation for this class was generated from the following file:

- SPService.h

5.103 SimpleMap Class Reference

The documentation for this class was generated from the following file:

- simplemap.h

5.104 StagingConfig Class Reference

Represents configuration of DTR data staging.

```
#include <conf_staging.h>
```

Public Member Functions

- StagingConfig (const GMEnvironment &env)

Friends

- class DTRGenerator

5.104.1 Detailed Description

Represents configuration of DTR data staging.

5.104.2 Constructor & Destructor Documentation

5.104.2.1 StagingConfig::StagingConfig (const GMEnvironment & *env*)

Load config from configuration file. Information from JobsListConfig is used first, then it is overwritten by parameters in [data-staging] (for ini style) or new staging parameters in <dataTransfer> (for xml style).

The documentation for this class was generated from the following file:

- conf_staging.h

5.105 DREService::Task Class Reference

Public Member Functions

- Task (int taskID, Arc::Message ∗request, Arc::Message ∗response)
- virtual ∼Task (void)

5.105.1 Constructor & Destructor Documentation

5.105.1.1 DREService::Task::Task (int *taskID*, Arc::Message ∗ *request*, Arc::Message ∗ *response*)

Constructor which is capable to extract prefix and suffix for the echo service.

5.105.1.2 virtual DREService::Task::∼Task (void) `[virtual]`

Destructor.

The documentation for this class was generated from the following file:

- Task.h

5.106 DREService::TaskQueue Class Reference

Public Member Functions

- TaskQueue (int length)
- virtual ∼TaskQueue (void)
- int pushTask (Task ∗task)
- Task ∗ shiftTask ()

5.106.1 Constructor & Destructor Documentation

5.106.1.1 DREService::TaskQueue::TaskQueue (int *length*)

Constructor which is capable to extract prefix and suffix for the echo service.

5.106.1.2 virtual DREService::TaskQueue::∼TaskQueue (void) `[virtual]`

Destructor.

5.106.2 Member Function Documentation

5.106.2.1 int DREService::TaskQueue::pushTask (Task ∗ *task*)

Blocks, if taskqueue is full. If task is stored in the queue and had a taskID == -1 it gets a fresh taskID.

5.106.2.2 Task∗ DREService::TaskQueue::shiftTask ()

Shifts the first task from the queue (and removes it).

The documentation for this class was generated from the following file:

- TaskQueue.h

5.107 DREService::TaskSet Class Reference

Public Member Functions

- TaskSet (int size)
- virtual ∼TaskSet (void)
- Task ∗ removeTask (int)

5.107.1 Constructor & Destructor Documentation

5.107.1.1 DREService::TaskSet::TaskSet (int *size*)

Constructor which is capable to extract prefix and suffix for the echo service.

5.107.1.2 virtual DREService::TaskSet::∼TaskSet (void) `[virtual]`

Destructor.

5.107.2 Member Function Documentation

5.107.2.1 Task∗ DREService::TaskSet::removeTask (int)

Checks wheter there is a task in the queue having that taskID in order to return it. If such a taskID is not available, themethod blocks until such a taskID is available. The task will be removed from the stack in that case.

The documentation for this class was generated from the following file:

- TaskSet.h

5.108 UnixMap Class Reference

Data Structures

- struct **source_t**
- class **unix_user_t**

The documentation for this class was generated from the following file:

- unixmap.h

5.109 gridftpd::UrlMapConfig Class Reference

The documentation for this class was generated from the following file:

- gridftpd/conf/conf_map.h

5.110 UrlMapConfig Class Reference

The documentation for this class was generated from the following file:

- a-rex/grid-manager/conf/conf_map.h

5.111 userspec_t Class Reference

The documentation for this class was generated from the following file:

- userspec.h

5.112 value_for_shell Class Reference

The documentation for this class was generated from the following file:

- job_desc.h

5.113 voms Struct Reference

```
#include <auth.h>
```

Data Fields

- std::string server
- std::string voname
- std::vector< voms_attrs > attrs

5.113.1 Detailed Description

VOMS data

5.113.2 Field Documentation

5.113.2.1 std::vector<voms_attrs> voms::attrs

User's characteristics

5.113.2.2 std::string voms::server

The VOMS server DN, as from its certificate

5.113.2.3 std::string voms::voname

The name of the VO to which the VOMS belongs

The documentation for this struct was generated from the following file:

- auth.h

5.114 voms_attrs Struct Reference

```
#include <auth.h>
```

Data Fields

- std::string group
- std::string role
- std::string cap

5.114.1 Detailed Description

VOMS attributes

5.114.2 Field Documentation

5.114.2.1 std::string voms_attrs::cap

user's capability

5.114.2.2 std::string voms_attrs::group

user's group

5.114.2.3 std::string voms_attrs::role

user's role

The documentation for this struct was generated from the following file:

- auth.h

5.115 ZeroUInt Class Reference

```
#include <job_config.h>
```

5.115.1 Detailed Description

ZeroUInt is a wrapper around unsigned int. It provides a consistent default value, as int type variables have no predefined value assigned upon creation. It also protects from potential counter underflow, to stop counter jumping to MAX_INT.

The documentation for this class was generated from the following file:

- job_config.h

Index