Hosting Environment (Daemon) Services

Generated by Doxygen 1.7.1

Fri Oct 19 2012 12:25:48

Contents

Chapter 1

Data Structure Index

1.1 Class Hierarchy

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

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

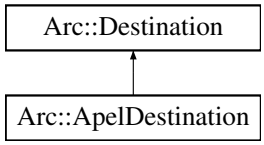
Chapter 3

Data Structure Documentation

3.1 Arc::ApelDestination Class Reference

`#include <ApelDestination.h>`

Inheritance diagram for Arc::ApelDestination:



Public Member Functions

- ApelDestination (JobLogFile &joblog)
- void report (JobLogFile &joblog)
- void finish ()

3.1.1 Detailed Description

Reporting destination adapter for APEL.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Arc::ApelDestination::ApelDestination (JobLogFile & *joblog*)

Constructor. Service URL and APEL-related parameters (e.g. UR batch size) are extracted from the given job log file.

3.1.3 Member Function Documentation

3.1.3.1 void Arc::ApelDestination::finish ( ) [virtual]

Finishes pending submission of records.

Reimplemented from Arc::Destination.

3.1.3.2 void Arc::ApelDestination::report (JobLogFile & *joblog*) [virtual]

Generates record from job log file content, collects it into the UR batch, and if batch is full, submits it to the service.

Implements Arc::Destination.

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

- ApelDestination.h

3.2 ARex::ARexGMConfig Class Reference

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

- job.h

3.3 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)

3.3.1 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.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 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

3.3.2.2 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

3.3.3 Member Function Documentation

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

Cancel processing/execution of job

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

Select a session dir to use for this job

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

Remove job from local pool

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

Returns time when job was created.

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

Creates file in job's session directory and returns handler

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

Returns true if job has failed

3.3.3.7 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.

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

Returns textual description of failure of last operation

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

Fills provided jsdl with job description

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

Return ID assigned to job

3.3.3.11 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.

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

Returns name of virtual log directory

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

Returns list of existing log files

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

Returns time when job state was last modified.

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

Opens directory inside session directory

3.3.3.16 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

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

Opens log file in control directory

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

Resume execution of job after error

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

Returns path to session directory

3.3.3.20 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

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

Returns current state of job

3.3.3.22 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.

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

Updates job credentials

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

- job.h

3.4 ARex::ARexService Class Reference

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

- arex.h

3.5 AuthEvaluator Class Reference

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

- auth.h

3.6 AuthUser Class Reference

Data Structures

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

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

- auth.h

3.7 AuthVO Class Reference

Friends

- class AuthUser

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

- auth.h

3.8 CacheConfig Class Reference

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

Public Member Functions

- CacheConfig (const GMEnvironment &env, std::string username="")
- CacheConfig (Arc::XMLNode cfg, std::string username="")
- CacheConfig ()
- void setCacheDirs (std::vector< std::string > cache_dirs)

3.8.1 Detailed Description

Reads conf file and provides methods to obtain cache info from it. Methods of this class may throw CacheConfigException.

3.8.2 Constructor & Destructor Documentation

3.8.2.1 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.

3.8.2.2 CacheConfig::CacheConfig (Arc::XMLNode *cfg*, std::string *username* = "")

Create a new CacheConfig instance. Read the XML config tree and fill in private member variables with cache parameters. If cfg points to parent node of <control> elements then all <control> elements are checked for user matching specified username. Any username matches "." in configuration tree. If cfg corresponds to <control> element then this element is used without performing matching. Cache parameters of the matched user are used for filling internal variables.

3.8.2.3 CacheConfig::CacheConfig ( ) [inline]

Empty CacheConfig

3.8.3 Member Function Documentation

3.8.3.1 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

3.9 CacheConfigException Class Reference

`#include <conf_cache.h>`

3.9.1 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

3.10 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)
- Arc::MCC_Status CacheLinkQuery (Arc::XMLNode in, Arc::XMLNode out)

3.10.1 Detailed Description

CacheService provides functionality for A-REX cache operations that can be performed by remote clients. It currently consists of three 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. CacheLinkQuery - query the status of a transfer initiated by CacheLink. This service 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 DTR data staging framework.

3.10.2 Constructor & Destructor Documentation

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

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

3.10.2.2 virtual Cache::CacheService::~CacheService (void) [virtual]

Destroy the CacheService

3.10.3 Member Function Documentation

3.10.3.1 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

3.10.3.2 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 data staging requests are entered. The user should then use CacheLinkQuery to poll the status of the requests.

Parameters

user A-REX user configuration for the mapped user

mapped_user The local user to which the client DN was mapped

3.10.3.3 Arc::MCC_Status Cache::CacheService::CacheLinkQuery (Arc::XMLNode *in*, Arc::XMLNode *out*) [protected]

Query the status of data staging for a given job ID.

3.10.3.4 Cache::CacheService::operator bool (void) [inline]

Returns true if the CacheService is valid.

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

Returns true if the CacheService is not valid.

3.10.3.6 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.

3.10.3.7 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

3.11 Cache::CacheServiceGenerator Class Reference

DTR Generator for the cache service.

`#include <CacheServiceGenerator.h>`

Public Member Functions

- CacheServiceGenerator (const JobUsers &users, bool with_arex)
- ~CacheServiceGenerator ()
- void receiveDTR (DataStaging::DTR_ptr dtr)
- bool addNewRequest (const JobUser &jobuser, const std::string &source, const std::string &destination, const Arc::UserConfig &usercfg, const std::string &jobid, uid_t uid, int priority)
- bool queryRequestsFinished (const std::string &jobid, std::string &error)

3.11.1 Detailed Description

DTR Generator for the cache service.

3.11.2 Constructor & Destructor Documentation

3.11.2.1 Cache::CacheServiceGenerator::CacheServiceGenerator (const JobUsers & *users,* bool *with_arex*)

Start Generator and get Scheduler instance.

If with_arex is true then it is assumed that A-REX takes care of configuring, starting and stopping the DTR Scheduler. If cache service is run outside of A-REX then it starts an independent DTR instance, using parameters given in arc.conf.

Parameters

users JobUsers object for A-REX configuration information

with_arex If true then we assume A-REX starts the scheduler, if false then we start and stop it.

3.11.3 Member Function Documentation

3.11.3.1 bool Cache::CacheServiceGenerator::addNewRequest (const JobUser & *jobuser,* const std::string & *source,* const std::string & *destination,* const Arc::UserConfig & *usercfg,* const std::string & *jobid,* uid_t *uid,* int *priority*)

Add a new request.

Parameters

jobuser JobUser for this transfer

source Source file

destination Destination file

usercfg UserConfig with proxy information

jobid Job identifier

uid uid under which to access session dir

priority DTR priority

3.11.3.2 bool Cache::CacheServiceGenerator::queryRequestsFinished (const std::string & *jobid*, std::string & *error*)

Query requests for given job id.

Parameters

jobid Job ID to query

error If any DTR finished with an error, the description is put in error.

Returns

True if all requests for the job have finished, false otherwise

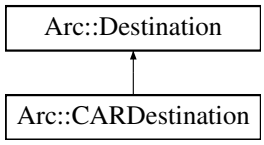
The documentation for this class was generated from the following file:

- CacheServiceGenerator.h

3.12 Arc::CARDestination Class Reference

`#include <CARDestination.h>`

Inheritance diagram for Arc::CARDestination:



Public Member Functions

- CARDestination (JobLogFile &joblog)
- void report (JobLogFile &joblog)
- void finish ()

3.12.1 Detailed Description

Reporting destination adapter for EMI.

3.12.2 Constructor & Destructor Documentation

3.12.2.1 Arc::CARDestination::CARDestination (JobLogFile & *joblog*)

Constructor. Service URL and CAR-related parameters (e.g. UR batch size) are extracted from the given job log file.

3.12.3 Member Function Documentation

3.12.3.1 void Arc::CARDestination::finish ( ) [virtual]

Finishes pending submission of records.

Reimplemented from Arc::Destination.

3.12.3.2 void Arc::CARDestination::report (JobLogFile & *joblog*) [virtual]

Generates record from job log file content, collects it into the UR batch, and if batch is full, submits it to the service.

Implements Arc::Destination.

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

- CARDestination.h

3.13 CommFIFO Class Reference

Data Structures

- class **elem_t**

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

- commfifo.h

3.14 gridftpd::ConfigSections Class Reference

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

- gridftpd/conf/conf_sections.h

3.15 ConfigSections Class Reference

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

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

3.16 ContinuationPlugins Class Reference

Data Structures

- class **command_t**
- class result_t

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

- plugins.h

3.17 ARex::CountedResource Class Reference

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

- arex.h

3.18 gridftpd::Daemon Class Reference

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

- daemon.h

3.19 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)

3.19.1 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

3.20 ARex::DelegationStore Class Reference

Data Structures

- class **Consumer**

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

- DelegationStore.h

3.21 ARex::DelegationStores Class Reference

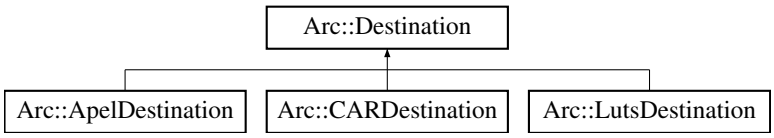
The documentation for this class was generated from the following file:

- DelegationStores.h

3.22 Arc::Destination Class Reference

`#include <Destination.h>`

Inheritance diagram for Arc::Destination:



Public Member Functions

- virtual void report (Arc::JobLogFile &joblog)=0
- virtual void finish ()

Static Public Member Functions

- static Destination * createDestination (Arc::JobLogFile &joblog)

3.22.1 Detailed Description

Abstract class to represent a reporting destination. Specific destination types are represented by inherited classes.

3.22.2 Member Function Documentation

3.22.2.1 static Destination* Arc::Destination::createDestination (Arc::JobLogFile & *joblog*) [static]

Creates an instance of the inherited class corresponding to the destination for the given job log file.

3.22.2.2 virtual void Arc::Destination::finish ( ) [inline, virtual]

Finishes pending submission of records.

Reimplemented in Arc::ApelDestination, Arc::CARDestination, and Arc::LutsDestination.

3.22.2.3 virtual void Arc::Destination::report (Arc::JobLogFile & *joblog*) [pure virtual]

Reports the job log file content to the destination.

Implemented in Arc::ApelDestination, Arc::CARDestination, and Arc::LutsDestination.

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

- Destination.h

3.23 Arc::Destinations Class Reference

`#include <Destinations.h>`

Public Member Functions

- void report (Arc::JobLogFile &joblog)

3.23.1 Detailed Description

Class to handle a set of reporting destinations.

3.23.2 Member Function Documentation

3.23.2.1 void Arc::Destinations::report (Arc::JobLogFile & *joblog*)

Reports the given job log file to a destination. If an adapter object for the specific destination already exists in the set, it uses that, otherwise creates a new one.

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

- Destinations.h

3.24 DirectAccess::diraccess_t Struct Reference

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

- fileplugin.h

3.25 DirectAccess Class Reference

Data Structures

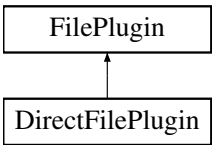
- struct diraccess_t

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

- fileplugin.h

3.26 DirectFilePlugin Class Reference

Inheritance diagram for DirectFilePlugin:



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

- fileplugin.h

3.27 DirEntry Class Reference

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

- fileroot.h

3.28 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)

3.28.1 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.

3.28.2 Constructor & Destructor Documentation

3.28.2.1 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

3.28.2.2 DTRGenerator::~DTRGenerator ( )

Stop Generator

3.28.3 Member Function Documentation

3.28.3.1 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

3.28.3.2 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

3.28.3.3 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.

3.28.3.4 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.

3.28.3.5 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

3.28.3.6 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.

3.28.3.7 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

3.29 DTRInfo Class Reference

`#include <dtr_generator.h>`

Public Member Functions

- DTRInfo (const JobUsers &users)

3.29.1 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.

3.29.2 Constructor & Destructor Documentation

3.29.2.1 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

3.30 Entry Class Reference

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

- Entry.h

3.31 Exec Class Reference

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

- info_types.h

3.32 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)

3.32.1 Detailed Description

Representation of delivered file chunks.

3.32.2 Member Function Documentation

3.32.2.1 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.

3.32.2.2 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

3.33 ARex::FileChunksList Class Reference

Container for FileChunks instances.

`#include <FileChunks.h>`

Public Member Functions

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

3.33.1 Detailed Description

Container for FileChunks instances.

3.33.2 Member Function Documentation

3.33.2.1 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

3.34 ARex::FileChunksRef Class Reference

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

- FileChunks.h

3.35 FileData Class Reference

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

- info_types.h

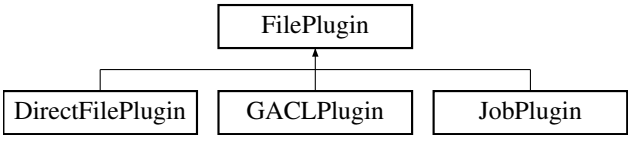
3.36 FileNode Class Reference

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

- fileroot.h

3.37 FilePlugin Class Reference

Inheritance diagram for FilePlugin:



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

- fileroot.h

3.38 ARex::FileRecord Class Reference

Data Structures

- class Iterator

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

- FileRecord.h

3.39 FileRoot Class Reference

Data Structures

- class ServerParams

Friends

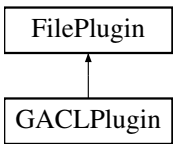
- class GridFTP_Commands

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

- fileroot.h

3.40 GACLPlugin Class Reference

Inheritance diagram for GACLPlugin:



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

- gaclplugin.h

3.41 gm_dirs_ Struct Reference

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

- jobplugin.h

3.42 GMEnvironment Class Reference

Public Member Functions

- std::string nordugrid_config_loc (void) const
- std::string support_mail_address (void) const
- std::string scratch_dir () const

3.42.1 Member Function Documentation

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

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

3.42.1.2 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

3.43 gridftpd::GMEnvironment Class Reference

Public Member Functions

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

3.43.1 Member Function Documentation

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

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

3.43.1.2 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

3.44 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

3.45 GridFTP_Commands_timeout Class Reference

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

- commands.h

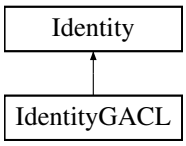
3.46 ARex::GridManager Class Reference

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

- grid_manager.h

3.47 Identity Class Reference

Inheritance diagram for Identity:



Data Structures

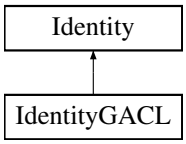
- class Item

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

- identity.h

3.48 IdentityGACL Class Reference

Inheritance diagram for IdentityGACL:

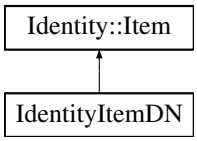


The documentation for this class was generated from the following file:

- identity_gacl.h

3.49 IdentityItemDN Class Reference

Inheritance diagram for IdentityItemDN:

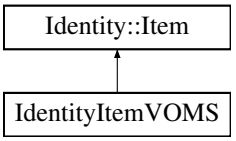


The documentation for this class was generated from the following file:

- identity_dn.h

3.50 IdentityItemVOMS Class Reference

Inheritance diagram for IdentityItemVOMS:



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

- identity_voms.h

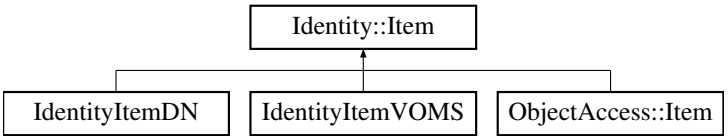
3.51 Index Class Reference

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

- Index.h

3.52 Identity::Item Class Reference

Inheritance diagram for Identity::Item:

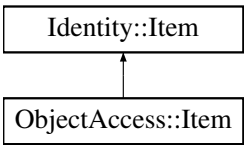


The documentation for this class was generated from the following file:

- identity.h

3.53 ObjectAccess::Item Class Reference

Inheritance diagram for ObjectAccess::Item:



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

- object_access.h

3.54 ARex::FileRecord::Iterator Class Reference

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

- FileRecord.h

3.55 job_state_rec_t Struct Reference

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

- grid-manager/jobs/job.h

3.56 JobDescription Class Reference

Friends

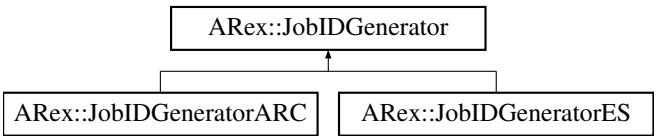
- class JobsList

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

- grid-manager/jobs/job.h

3.57 ARex::JobIDGenerator Class Reference

Inheritance diagram for ARex::JobIDGenerator:

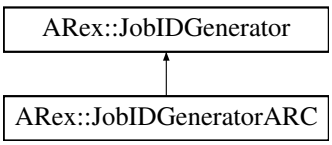


The documentation for this class was generated from the following file:

- tools.h

3.58 ARex::JobIDGeneratorARC Class Reference

Inheritance diagram for ARex::JobIDGeneratorARC:

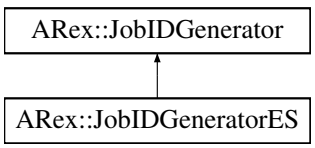


The documentation for this class was generated from the following file:

- tools.h

3.59 ARex::JobIDGeneratorES Class Reference

Inheritance diagram for ARex::JobIDGeneratorES:



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

- tools.h

3.60 JobLocalDescription Class Reference

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

- info_types.h

3.61 JobLog Class Reference

`#include <job_log.h>`

3.61.1 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

3.62 Arc::JobLogFile Class Reference

`#include <JobLogFile.h>`

Public Member Functions

- JobLogFile (const std::string &_filename)
- int parse (const std::string &_filename)
- void createUsageRecord (Arc::XMLNode &usagerecord, const char *recordid_prefix="ur-")
- void createCARUsageRecord (Arc::XMLNode &usagerecord, const char *recordid_prefix="ur-")
- std::string getFilename ()
- void allowRemove (bool a)
- bool exists ()
- bool olderThan (time_t age)
- void remove ()

3.62.1 Detailed Description

Class to represent a job log file created by A-REX, and to create OGF Job Usage Records from them.

3.62.2 Constructor & Destructor Documentation

3.62.2.1 Arc::JobLogFile::JobLogFile (const std::string & *_filename*) [inline]

Constructor. Loads and parses A-REX job log.

References parse().

3.62.3 Member Function Documentation

3.62.3.1 void Arc::JobLogFile::allowRemove (bool *a*) [inline]

Enables/disables file removal from disk

3.62.3.2 void Arc::JobLogFile::createCARUsageRecord (Arc::XMLNode & *usagerecord,* const char * *recordid_prefix* = "ur-")

Creates an OGF 2.0 (CAR) Job Usage Record from parsed log files.

3.62.3.3 void Arc::JobLogFile::createUsageRecord (Arc::XMLNode & *usagerecord,* const char * *recordid_prefix* = "ur-")

Creates an OGF Job Usage Record from parsed log files.

- Missing UR properties:
  1. ProcessID: Local PID(s) of job. Extraction is LRMS-specific and may not always be possible
  2. Charge: Amount of money or abstract credits charged for the job.
  3. Some differentiated properties e.g. network, disk etc.

3.62.3.4 bool Arc::JobLogFile::exists ( )

Checks if file exists on the disk

3.62.3.5 std::string Arc::JobLogFile::getFilename ( ) [inline]

Returns original full path to log file

3.62.3.6 bool Arc::JobLogFile::olderThan (time_t *age*)

Checks if file was modified earlier than 'age' seconds ago

3.62.3.7 int Arc::JobLogFile::parse (const std::string & *_filename*)

Reloads and parses A-REX job log.

Referenced by JobLogFile().

3.62.3.8 void Arc::JobLogFile::remove ( )

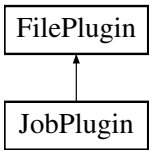
Deletes file from the disk

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

- JobLogFile.h

3.63 JobPlugin Class Reference

Inheritance diagram for JobPlugin:



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

- jobplugin.h

3.64 JobsList Class Reference

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

- states.h

3.65 JobsListConfig Class Reference

`#include <job_config.h>`

Friends

- class JobsList

3.65.1 Detailed Description

Class to represent information read from configuration.

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

- job_config.h

3.66 JobUser Class Reference

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

- users.h

3.67 JobUserHelper Class Reference

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

- users.h

3.68 JobUsers Class Reference

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

- users.h

3.69 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 ()

3.69.1 Detailed Description

LdapQuery class; querying of LDAP servers.

3.69.2 Member Enumeration Documentation

3.69.2.1 enum gridftpd::LdapQuery::Scope

Scope for a LDAP queries. Use when querying.

3.69.3 Constructor & Destructor Documentation

3.69.3.1 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.

3.69.3.2 gridftpd::LdapQuery::~LdapQuery ( )

Destructor. Will disconnect from the ldapserver if stll connected.

3.69.4 Member Function Documentation

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

Returns the hostname of the ldap-server.

3.69.4.2 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.

3.69.4.3 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

3.70 gridftpd::LdapQueryError Class Reference

`#include <ldapquery.h>`

Public Member Functions

- LdapQueryError (std::string message)

3.70.1 Detailed Description

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

3.70.2 Constructor & Destructor Documentation

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

Standard exception class constructor.

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

- ldapquery.h

3.71 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

3.72 gridftpd::RunPlugin::lib_plugin_t Union Reference

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

- gridftpd/run/run_plugin.h

3.73 LRMSResult Class Reference

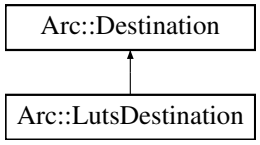
The documentation for this class was generated from the following file:

- info_types.h

3.74 Arc::LutsDestination Class Reference

`#include <LutsDestination.h>`

Inheritance diagram for Arc::LutsDestination:



Public Member Functions

- LutsDestination (JobLogFile &joblog)
- void report (JobLogFile &joblog)
- void finish ()

3.74.1 Detailed Description

Reporting destination adapter for SGAS LUTS.

3.74.2 Constructor & Destructor Documentation

3.74.2.1 Arc::LutsDestination::LutsDestination (JobLogFile & *joblog*)

Constructor. Service URL and LUTS-related parameters (e.g. UR batch size) are extracted from the given job log file.

3.74.3 Member Function Documentation

3.74.3.1 void Arc::LutsDestination::finish ( ) [virtual]

Finishes pending submission of records.

Reimplemented from Arc::Destination.

3.74.3.2 void Arc::LutsDestination::report (JobLogFile & *joblog*) [virtual]

Generates record from job log file content, collects it into the UR batch, and if batch is full, submits it to the service.

Implements Arc::Destination.

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

- LutsDestination.h

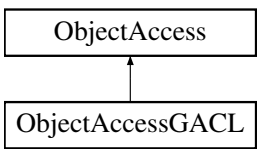
3.75 numvalue_for_shell Class Reference

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

- job_desc.h

3.76 ObjectAccess Class Reference

Inheritance diagram for ObjectAccess:



Data Structures

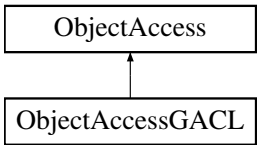
- class Item

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

- object_access.h

3.77 ObjectAccessGACL Class Reference

Inheritance diagram for ObjectAccessGACL:



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

- object_access_gacl.h

3.78 ARex::OptimizedInformationContainer Class Reference

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

- arex.h

3.79 gridftpd::ParallelLdapQueries Class Reference

`#include <ldapquery.h>`

3.79.1 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

3.80 ARex::PayloadBigFile Class Reference

Public Member Functions

- PayloadBigFile (const char *filename, Size_t start, Size_t end)
- virtual ~PayloadBigFile (void)

3.80.1 Constructor & Destructor Documentation

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

Creates object associated with file for reading from it

3.80.1.2 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:

- PayloadFile.h

3.81 ARex::PayloadFAFile Class Reference

Public Member Functions

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

3.81.1 Constructor & Destructor Documentation

3.81.1.1 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:

- PayloadFile.h

3.82 ARex::PayloadFile Class Reference

`#include <PayloadFile.h>`

Public Member Functions

- PayloadFile (const char *filename, Size_t start, Size_t end)
- virtual ~PayloadFile (void)

3.82.1 Detailed Description

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

3.82.2 Constructor & Destructor Documentation

3.82.2.1 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.

3.82.2.2 virtual ARex::PayloadFile::~PayloadFile (void) [virtual]

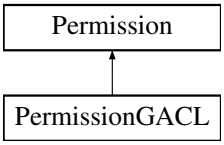
Creates object associated with file for writing into it. Use size=-1 for undefined size.

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

- PayloadFile.h

3.83 Permission Class Reference

Inheritance diagram for Permission:

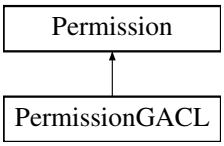


The documentation for this class was generated from the following file:

- permission.h

3.84 PermissionGACL Class Reference

Inheritance diagram for PermissionGACL:



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

- permission_gacl.h

3.85 Policy Class Reference

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

- Policy.h

3.86 ContinuationPlugins::result_t Class Reference

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

- plugins.h

3.87 RunParallel Class Reference

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

- run_parallel.h

3.88 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

3.89 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

3.90 RunPlugins Class Reference

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

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

3.91 RunRedirected Class Reference

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

- run_redirected.h

3.92 Server Class Reference

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

- Server.h

3.93 FileRoot::ServerParams Class Reference

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

- fileroot.h

3.94 ArcSec::Service_AA Class Reference

`#include <aaservice.h>`

3.94.1 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

3.95 Arc::Service_JavaWrapper Class Reference

Public Member Functions

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

3.95.1 Member Function Documentation

3.95.1.1 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

3.96 Arc::Service_PythonWrapper Class Reference

Public Member Functions

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

3.96.1 Member Function Documentation

3.96.1.1 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

3.97 ArcSec::Service_SLCS Class Reference

`#include <slcs.h>`

3.97.1 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

3.98 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 &)

3.98.1 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

3.98.2 Constructor & Destructor Documentation

3.98.2.1 SPService::Service_SP::Service_SP (Arc::Config * *cfg*)

Constructor

3.98.3 Member Function Documentation

3.98.3.1 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

3.99 SimpleMap Class Reference

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

- simplemap.h

3.100 StagingConfig Class Reference

Represents configuration of DTR data staging.

`#include <conf_staging.h>`

Public Member Functions

- StagingConfig (const GMEnvironment &env)

Friends

- class DTRGenerator

3.100.1 Detailed Description

Represents configuration of DTR data staging.

3.100.2 Constructor & Destructor Documentation

3.100.2.1 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

3.101 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

3.102 UrlMapConfig Class Reference

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

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

3.103 gridftpd::UrlMapConfig Class Reference

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

- gridftpd/conf/conf_map.h

3.104 Arc::UsageReporter Class Reference

```
#include <UsageReporter.h>
```

Public Member Functions

- UsageReporter (std::string job_log_dir_, time_t expiration_time_=0, std::vector< std::string > urls_=std::vector< std::string >(), std::vector< std::string > topics_=std::vector< std::string >(), std::string out_dir_="")
- int report ()

3.104.1 Detailed Description

The class for main JURA functionality. Traverses the 'logs' dir of the given control directory, and reports usage data extracted from job log files within.

3.104.2 Constructor & Destructor Documentation

3.104.2.1 Arc::UsageReporter::UsageReporter (std::string *job_log_dir_*, time_t *expiration_time_* = *0*, std::vector< std::string > *urls_* = `std::vector< std::string >()`, std::vector< std::string > *topics_* = `std::vector< std::string >()`, std::string *out_dir_* = `""`)

Constructor. Gets the job log dir and the expiration time in seconds. Default expiration time is infinity (represented by zero value).

3.104.3 Member Function Documentation

3.104.3.1 int Arc::UsageReporter::report ( )

Processes job log files in '<control_dir>/logs'.

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

- UsageReporter.h

3.105 userspec_t Class Reference

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

- userspec.h

3.106 value_for_shell Class Reference

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

- job_desc.h

3.107 voms Struct Reference

`#include <auth.h>`

Data Fields

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

3.107.1 Detailed Description

VOMS data

3.107.2 Field Documentation

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

User's characteristics

3.107.2.2 std::string voms::server

The VOMS server DN, as from its certificate

3.107.2.3 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

3.108 voms_attrs Struct Reference

`#include <auth.h>`

Data Fields

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

3.108.1 Detailed Description

VOMS attributes

3.108.2 Field Documentation

3.108.2.1 std::string voms_attrs::cap

user's capability

3.108.2.2 std::string voms_attrs::group

user's group

3.108.2.3 std::string voms_attrs::role

user's role

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

- auth.h

3.109 ZeroUInt Class Reference

`#include <job_config.h>`

3.109.1 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