Hosting Environment (Daemon) Services Reference Manual

Generated by Doxygen 1.4.7

Mon Feb 2 23:22:34 2009

Contents

Chapter 1

Hosting Environment (Daemon) Services Hierarchical Index

1.1 Hosting Environment (Daemon) Services Class Hierarchy

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

Chapter 2

Hosting Environment (Daemon) Services Class Index

2.1 Hosting Environment (Daemon) Services Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Hosting Environment (Daemon) Services File Index

3.1 Hosting Environment (Daemon) Services File List

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

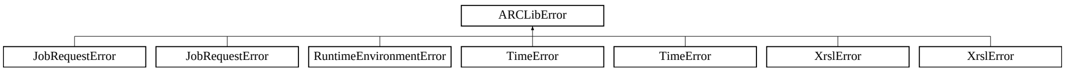
Chapter 4

Hosting Environment (Daemon) Services Class Documentation

4.1 ARCLibError Class Reference

```
#include <error.h>
```

Inheritance diagram for ARCLibError::



Public Member Functions

- ARCLibError (std::string message)
- virtual ∼ARCLibError () throw ()
- virtual const char ∗ what () const throw ()

4.1.1 Detailed Description

This is the top exception for ARCLib. Every exeption in ARCLib should inherit from this. The exception inherits from the top C++ exception: std::exception.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 ARCLibError::ARCLibError (std::string *message*) `[inline]`

Creates a new exception, with the mesage given as argument

4.1.2.2 virtual ARCLibError::∼ARCLibError () throw () `[inline, virtual]`

Destructor. Not that much to say.

4.1.3 Member Function Documentation

4.1.3.1 virtual const char* ARCLibError::what (void) const throw () `[inline, virtual]`

Returns the message given in the constructer.

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

- error.h

4.2 Arc::AREXClient Class Reference

A client class for the A-REX service.

```
#include <arex_client.h>
```

Public Member Functions

- AREXClient (std::string configFile="") throw (AREXClientError)
- **AREXClient** (const Arc::URL &url, const Arc::MCCConfig &cfg) throw (AREXClientError)
- ∼AREXClient ()
- std::string submit (std::istream &jsdl_file, AREXFileList &file_list, bool delegate=false) throw (AREXClientError)
- std::string stat (const std::string &jobid) throw (AREXClientError)
- void kill (const std::string &jobid) throw (AREXClientError)
- void clean (const std::string &jobid) throw (AREXClientError)
- std::string sstat (void) throw (AREXClientError)
- ClientSOAP ∗ **SOAP** (void)

4.2.1 Detailed Description

A client class for the A-REX service.

This class is a client for the A-REX service (Arc Resource-coupled EXecution service). It provides methods for three operations on an A-REX service:

- Job submission
- Job status queries
- Job termination

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Arc::AREXClient::AREXClient (std::string *configFile* = "") throw (AREXClientError)

The constructor for the AREXClient class.

This is the constructor for the AREXClient class. It creates an A-REX client that corresponds to a specific A-REX service, which is specified in a configuration file. The configuration file also specifies how to set up the communication chain for the client. The location of the configuration file can be provided as a parameter to the method. If no such parameter is given, the environment variable ARC_AREX_CONFIG is assumed to contain the location. If there is no such environment variable, the configuration file is assumed to be "arex_client.xml" in the current working directory.

Parameters:

configFile The location of the configuration file.

Exceptions:

An AREXClientError object if an error occurs.

4.2.2.2 Arc::AREXClient::∼AREXClient ()

The destructor.

This is the destructor. It does what destructors usually do, cleans up...

4.2.3 Member Function Documentation

4.2.3.1 void Arc::AREXClient::clean (const std::string & *jobid*) throw (AREXClientError)

Removes a job.

This method sends a request to the A-REX service to remove a job from it's pool. If job is running it will be killed by service as well.

Parameters:

jobid The Job ID of the job to remove.

Exceptions:

An AREXClientError object if an error occurs.

4.2.3.2 void Arc::AREXClient::kill (const std::string & *jobid*) throw (AREXClientError)

Terminates a job.

This method sends a request to the A-REX service to terminate a job.

Parameters:

jobid The Job ID of the job to terminate.

Exceptions:

An AREXClientError object if an error occurs.

4.2.3.3 std::string Arc::AREXClient::sstat (void) throw (AREXClientError)

Query the status of a service.

This method queries the A-REX service about it's status.

Returns:

The XML document representing status of the service.

Exceptions:

An AREXClientError object if an error occurs.

4.2.3.4 std::string Arc::AREXClient::stat (const std::string & *jobid*) throw (AREXClientError)

Query the status of a job.

This method queries the A-REX service about the status of a job.

Parameters:

jobid The Job ID of the job.

Returns:

The status of the job.

Exceptions:

An AREXClientError object if an error occurs.

4.2.3.5 std::string Arc::AREXClient::submit (std::istream & *jsdl_file*, AREXFileList & *file_list*, bool *delegate* = `false`) throw (AREXClientError)

Submit a job.

This method submits a job to the A-REX service corresponding to this client instance.

Parameters:

jsdl_file An input stream from which the JSDL file for the job can be read.

Returns:

The Job ID of the the submitted job.

Exceptions:

An AREXClientError object if an error occurs.

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

- arex_client.h

4.3 Arc::AREXClientError Class Reference

An exception class for the AREXClient class.

```
#include <arex_client.h>
```

Public Member Functions

- AREXClientError (const std::string &what="")

4.3.1 Detailed Description

An exception class for the AREXClient class.

This is an exception class that is used to handle runtime errors discovered in the AREXClient class.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Arc::AREXClientError::AREXClientError (const std::string & *what* = "")

Constructor.

This is the constructor of the AREXClientError class.

Parameters:

what An explanation of the error.

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

- arex_client.h

4.4 ARex::ARexJob Class Reference

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

4.4.1 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.

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

- a-rex/job.h

4.5 ARex::ARexJob Class Reference

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

4.5.1 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.

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

- a-rex/job.h

4.6 CacheConfig Class Reference

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

Public Member Functions

- CacheConfig (std::string username="")
- std::list< std::string > **getCacheDirs** ()
- void setCacheDirs (std::list< std::string > cache_dirs)
- int **getCacheMax** ()
- int **getCacheMin** ()
- bool **cleanCache** ()
- bool **oldConf** ()
- CacheConfig (std::string username="")
- std::list< std::list< std::string > > **getCacheDirs** ()
- void setCacheDirs (std::list< std::list< std::string > > cache_dirs)
- int **getCacheMax** ()
- int **getCacheMin** ()
- bool **cleanCache** ()
- bool **oldConf** ()

4.6.1 Detailed Description

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

4.6.2 Constructor & Destructor Documentation

4.6.2.1 CacheConfig::CacheConfig (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.

4.6.2.2 CacheConfig::CacheConfig (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.

4.6.3 Member Function Documentation

4.6.3.1 void CacheConfig::setCacheDirs (std::list< std::list< std::string > > *cache_dirs*) [inline]

To allow for substitutions done during configuration

4.6.3.2 void CacheConfig::setCacheDirs (std::list< std::string > *cache_dirs*) [inline]

To allow for substitutions done during configuration

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

- grid-manager/conf/conf_cache.h
- local/grid-manager/conf/conf_cache.h

4.7 CacheConfigException Class Reference

`#include <conf_cache.h>`

Public Member Functions

- **CacheConfigException** (std::string desc="")
- std::string **what** ()
- **CacheConfigException** (std::string desc="")
- std::string **what** ()

4.7.1 Detailed Description

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

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

- grid-manager/conf/conf_cache.h
- local/grid-manager/conf/conf_cache.h

4.8 ArcSec::Charon Class Reference

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

Public Member Functions

- **Charon** (Arc::Config ∗cfg)
- virtual Arc::MCC_Status **process** (Arc::Message &inmsg, Arc::Message &outmsg)

Protected Member Functions

- Arc::MCC_Status **make_soap_fault** (Arc::Message &outmsg)

Protected Attributes

- Arc::NS **ns_**
- Arc::Logger **logger_**

4.8.1 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

4.9 ARex::Config Class Reference

```
#include <configcore.h>
```

4.9.1 Detailed Description

Core configuration class.

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

- config/configcore.h

4.10 ARex::Config Class Reference

```
#include <configcore.h>
```

4.10.1 Detailed Description

Core configuration class.

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

- config/configcore.h

4.11 ARex::ConfigError Class Reference

```
#include <configcore.h>
```

4.11.1 Detailed Description

Error configuration class.

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

- config/configcore.h

4.12 ARex::ConfigError Class Reference

```
#include <configcore.h>
```

4.12.1 Detailed Description

Error configuration class.

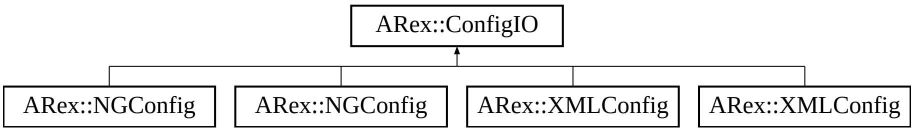
The documentation for this class was generated from the following file:

- config/configcore.h

4.13 ARex::ConfigIO Class Reference

```
#include <configio.h>
```

Inheritance diagram for ARex::ConfigIO::



4.13.1 Detailed Description

Virtual base-class for reading and writing configuration files. Concrete instances include NGConfig and XMLConfig.

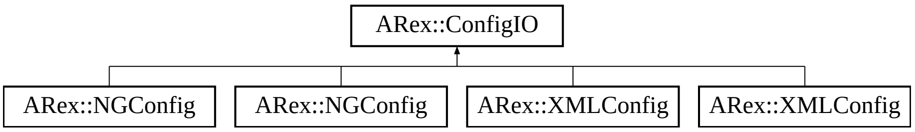
The documentation for this class was generated from the following file:

- config/configio.h

4.14 ARex::ConfigIO Class Reference

`#include <configio.h>`

Inheritance diagram for ARex::ConfigIO::



4.14.1 Detailed Description

Virtual base-class for reading and writing configuration files. Concrete instances include NGConfig and XMLConfig.

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

- config/configio.h

4.15 Job Class Reference

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

Public Member Functions

- Job (void)
- Job (std::string path)
- ~Job (void)
- operator bool (void)
- bool **operator!** (void)
- bool Start (void)
- bool Cancel (void)
- bool Resume (void)
- std::string GetState (void)
- std::string GetSessionDir (void)

4.15.1 Detailed Description

Collect all information (status, lrms info, user) required to handle job

4.15.2 Constructor & Destructor Documentation

4.15.2.1 Job::Job (void)

Constructor: Creates empty job

4.15.2.2 Job::Job (std::string *path*)

Constructor: load job information form files

4.15.2.3 Job::~Job (void)

Destuctior

4.15.3 Member Function Documentation

4.15.3.1 bool Job::Cancel (void)

Cancel processing/execution of job

4.15.3.2 std::string Job::GetSessionDir (void)

Returns the session directory of the job

4.15.3.3 std::string Job::GetState (void)

Returns the string represnetation of job state

4.15.3.4 Job::operator bool (void) `[inline]`

Helper logical operators

4.15.3.5 bool Job::Resume (void)

Resume execution of job after error

4.15.3.6 bool Job::Start (void)

Starts job. Most of the cases it means to submit to LRMS

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

- arex2/job.h

4.16 ARex2::JobControl Class Reference

```
#include <job_control.h>
```

Public Member Functions

- **JobControl** (JobUser &user, std::string &job_id)
- void **SetControlDir** (const std::string &dir)
- void **SetSessionDir** (const std::string &dir)
- const std::string & **ControlDir** (void) const
- const std::string & **SessionDir** (void) const
- bool **Serialize** (JobState &state)
- bool **Serialize** (JobDescription &desc)
- bool **Serialize** (JobUser &user)
- bool **Serialize** (Job &job)
- bool **DeSerialize** (JobState &state)
- bool **DeSerialize** (JobDescription &desc)
- bool **DeSerialize** (JobUser &user)
- bool **DeSerialize** (Job &job)

Protected Attributes

- std::string **control_dir_**
- std::string **session_dir_**

4.16.1 Detailed Description

Represents job controll information like session dir, control dir

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

- job_control.h

4.17 ARex2::JobDataCache Class Reference

```
#include <job_data_cache.h>
```

Public Member Functions

- **JobDataCache** (const std::string &dir, const std::string &data_dir, bool priv=false)
- **JobDataCache** (const std::string &dir, const std::string &data_dir, const std::string &link_dir, bool priv=false)
- void **SetCacheSize** (long long int cache_max, long long int cache_min=0)
- const std::string & **CacheDir** (void) const
- const std::string & **CacheDataDir** (void) const
- const std::string & **CacheLinkDir** (void) const
- long long int **CacheMaxSize** (void) const
- long long int **CacheMinSize** (void) const
- bool **CachePrivate** (void) const

4.17.1 Detailed Description

Data cache

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

- job_data_cache.h

4.18 ARex2::JobDescription Class Reference

```
#include <job_descr.h>
```

Public Member Functions

- **JobDescription** (const JobDescription &j)
- virtual JobDescription & **operator=** (const JobDescription &j) virtual ∼JobDescription(void)
- std::string & JobName (void)
- std::list< std::string > & **Arguments** (void)
- std::list< std::string > & **Executables** (void)
- std::list< RTE > & **RuntimeEnvironments** (void)
- std::list< RTE > & **Middlewares** (void)
- std::string & **ACL** (void)
- std::string & **GMLog** (void)
- std::list< std::string > & **Loggers** (void)
- std::string & **CredentialServer** (void)
- std::string & **Stdin** (void)
- std::string & **Stdout** (void)
- std::string & **Stderr** (void)
- std::string & **Queue** (void)
- std::list< Notification > & **Notifications** (void)
- long **LifeTime** (void)
- std::list< InputFile > & **InputData** (void)
- std::list< OutputFile > & **OutputData** (void)
- int **Memory** (void)
- long **CPUTime** (void)
- long **WallTime** (void)
- long **GridTime** (void)
- int **Count** (void)
- int **Reruns** (void)
- std::string & **ClientSoftware** (void)
- std::string & **ClientHostname** (void)
- int **Disk** (void)

Protected Attributes

- std::string **job_name**
- std::list< std::string > **arguments**
- std::list< std::string > **executables**
- std::list< RTE > **rtes**
- std::list< RTE > **middlewares**
- std::string **architecture**
- std::string **acl**
- Arc::Time **start_time**
- std::string **gmlog**
- std::list< std::string > **loggers**
- std::string **credentialserver**
- std::string **cluster**

- std::string **queue**
- std::string **sstdin**
- std::string **sstdout**
- std::string **sstderr**
- std::list< Notification > **notifications**
- long **lifetime**
- std::list< InputFile > **inputdata**
- std::list< OutputFile > **outputdata**
- int **memory**
- int **disk**
- long **cpu_time**
- long **wall_time**
- long **grid_time**
- int **count**
- int **reruns**
- std::string **client_software**
- std::string **client_hostname**

Classes

- class InputFile
- class Notification
- class OutputFile

4.18.1 Detailed Description

Internal representation of Job described by JSDL

4.18.2 Member Function Documentation

4.18.2.1 std::string& ARex2::JobDescription::JobName (void) `[inline]`

Interface methods to access stored values.

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

- job_descr.h

4.19 ARex2::JobDescription::InputFile Class Reference

```
#include <job_descr.h>
```

Public Member Functions

- **InputFile** (const std::string &name, const std::string &source)

Public Attributes

- std::string **name**
- std::string **parameters**
- Arc::URL **source**

4.19.1 Detailed Description

Class represents the one of the input file of the job

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

- job_descr.h

4.20 ARex2::JobDescription::Notification Class Reference

```
#include <job_descr.h>
```

Public Member Functions

- **Notification** (const std::string &flags, const std::string &email)

Public Attributes

- std::string **flags**
- std::string **email**

4.20.1 Detailed Description

Class represents notification requiest

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

- job_descr.h

4.21 ARex2::JobDescription::OutputFile Class Reference

```
#include <job_descr.h>
```

Public Member Functions

- **OutputFile** (const std::string &name, const std::string &destination)

Public Attributes

- std::string **name**
- Arc::URL **destination**

4.21.1 Detailed Description

Class represents the one of the output file of the job

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

- job_descr.h

4.22 JobLog Class Reference

`#include <job_log.h>`

Public Member Functions

- **JobLog** (const char ∗fname)
- void **SetOutput** (const char ∗fname)
- bool **start_info** (JobDescription &job, const JobUser &user)
- bool **finish_info** (JobDescription &job, const JobUser &user)
- bool **is_reporting** (void)
- bool **RunReporter** (JobUsers &users)
- bool **SetReporter** (const char ∗destination)
- void **SetExpiration** (time_t period=0)
- bool **make_file** (JobDescription &job, JobUser &user)
- **JobLog** (const char ∗fname)
- void **SetOutput** (const char ∗fname)
- bool **start_info** (JobDescription &job, const JobUser &user)
- bool **finish_info** (JobDescription &job, const JobUser &user)
- bool **is_reporting** (void)
- bool **RunReporter** (JobUsers &users)
- bool **SetReporter** (const char ∗destination)
- void **SetExpiration** (time_t period=0)
- bool **make_file** (JobDescription &job, JobUser &user)

Static Public Member Functions

- static bool **read_info** (std::fstream &i, bool &processed, bool &jobstart, struct tm &t, JobId &jobid, JobLocalDescription &job_desc, std::string &failure)
- static bool **read_info** (std::fstream &i, bool &processed, bool &jobstart, struct tm &t, JobId &jobid, JobLocalDescription &job_desc, std::string &failure)

4.22.1 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 files:

- grid-manager/log/job_log.h
- local/grid-manager/log/job_log.h

4.23 ARex2::JobLRMSInfo Class Reference

```
#include <lrms.h>
```

Protected Attributes

- std::string **id_**
- time_t **keep_finished**
- time_t **keep_deleted**

4.23.1 Detailed Description

Class represents the information about job in LRMS

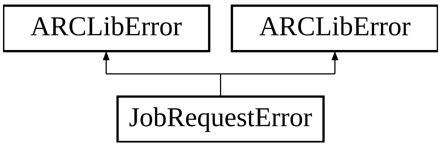
The documentation for this class was generated from the following file:

- lrms.h

4.24 JobRequestError Class Reference

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

Inheritance diagram for JobRequestError::



Public Member Functions

- JobRequestError (std::string message)
- JobRequestError (std::string message)

4.24.1 Detailed Description

Exception class thrown in case of errors with the JobRequest class.

4.24.2 Constructor & Destructor Documentation

4.24.2.1 JobRequestError::JobRequestError (std::string *message*) `[inline]`

Standard exception class constructor.

4.24.2.2 JobRequestError::JobRequestError (std::string *message*) `[inline]`

Standard exception class constructor.

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

- a-rex/grid-manager/jobdesc/job.h
- a-rex_local/grid-manager/jobdesc/job.h

4.25 JobRequestJSDL Class Reference

`#include <job_jsdl.h>`

Public Member Functions

- **JobRequestJSDL** (const JobRequest &j) throw (JobRequestError)
- **JobRequestJSDL** (const char ∗s) throw (JobRequestError)
- **JobRequestJSDL** (const std::string &s) throw (JobRequestError)
- **JobRequestJSDL** (std::istream &i) throw (JobRequestError)
- virtual JobRequest & **operator=** (const JobRequest &j) throw (JobRequestError)
- **JobRequestJSDL** (const JobRequest &j) throw (JobRequestError)
- **JobRequestJSDL** (const char ∗s) throw (JobRequestError)
- **JobRequestJSDL** (const std::string &s) throw (JobRequestError)
- **JobRequestJSDL** (std::istream &i) throw (JobRequestError)
- virtual JobRequest & **operator=** (const JobRequest &j) throw (JobRequestError)

Protected Member Functions

- bool **set** (std::istream &s) throw (JobRequestError)
- bool **set** (Arc::XMLNode jsdl_description_) throw (JobRequestError)
- double **get_limit** (Arc::XMLNode range)
- virtual bool **print** (std::string &s) throw (JobRequestError)
- bool **set** (std::istream &s) throw (JobRequestError)
- bool **set** (Arc::XMLNode jsdl_description_) throw (JobRequestError)
- double **get_limit** (Arc::XMLNode range)
- virtual bool **print** (std::string &s) throw (JobRequestError)

4.25.1 Detailed Description

Class to represent the request for computational job.

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

- grid-manager/jobdesc/job_jsdl.h
- local/grid-manager/jobdesc/job_jsdl.h

4.26 JobRequestXRSL Class Reference

```
#include <job_xrsl.h>
```

Public Types

- **UserFriendly**
- **NoUnits**
- enum **Type** { **UserFriendly**, **NoUnits** }
- enum **Type** { **UserFriendly**, **NoUnits** }

Public Member Functions

- **JobRequestXRSL** (const JobRequest &j, Type type=UserFriendly) throw (JobRequestError)
- **JobRequestXRSL** (const char ∗s, Type type=UserFriendly) throw (JobRequestError)
- **JobRequestXRSL** (const std::string &s, Type type=UserFriendly) throw (JobRequestError)
- **JobRequestXRSL** (std::istream &i, Type type=UserFriendly) throw (JobRequestError)
- virtual JobRequest & **operator=** (const JobRequest &j) throw (JobRequestError)
- **JobRequestXRSL** (const JobRequest &j, Type type=UserFriendly) throw (JobRequestError)
- **JobRequestXRSL** (const char ∗s, Type type=UserFriendly) throw (JobRequestError)
- **JobRequestXRSL** (const std::string &s, Type type=UserFriendly) throw (JobRequestError)
- **JobRequestXRSL** (std::istream &i, Type type=UserFriendly) throw (JobRequestError)
- virtual JobRequest & **operator=** (const JobRequest &j) throw (JobRequestError)

Protected Member Functions

- bool **set** (const char ∗s) throw (JobRequestError)
- bool **set** (Xrsl &xrsl) throw (JobRequestError)
- bool **set_xrsl** (Xrsl &xrsl) throw (JobRequestError)
- virtual bool **print** (std::string &s) throw (JobRequestError)
- bool **set** (const char ∗s) throw (JobRequestError)
- bool **set** (Xrsl &xrsl) throw (JobRequestError)
- bool **set_xrsl** (Xrsl &xrsl) throw (JobRequestError)
- virtual bool **print** (std::string &s) throw (JobRequestError)

4.26.1 Detailed Description

Class to represent the request for computational job.

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

- grid-manager/jobdesc/job_xrsl.h
- local/grid-manager/jobdesc/job_xrsl.h

4.27 ARex2::JobState Class Reference

```
#include <job_state.h>
```

Public Member Functions

- **JobState** (job_state_t s)
- const std::string & **GetFailure** (void)

Protected Attributes

- job_state_t **state**
- std::string **failure_reason**

4.27.1 Detailed Description

Represents the state of job. It includes error messages as well

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

- job_state.h

4.28 ARex2::JobUser Class Reference

```
#include <job_user.h>
```

Public Member Functions

- JobsList ∗ **get_jobs** () const
- void **operator=** (JobsList ∗jobs_list)
- **JobUser** (const std::string &unix_name, RunPlugin ∗cred_plugin=NULL)
- **JobUser** (uid_t uid, RunPlugin ∗cred_plugin=NULL)
- **JobUser** (const JobUser &user)
- void **SetKeepFinished** (time_t ttl)
- void **SetKeepDeleted** (time_t ttr)
- void **SetReruns** (int n)
- void **SetDiskSpace** (unsigned long long int n)
- void **SetStrictSession** (bool v)
- void **SetShareLevel** (jobinfo_share_t s)
- bool **CreateDirectories** (void)
- time_t **KeepFinished** (void) const
- time_t **KeepDeleted** (void) const
- bool **StrictSession** (void) const
- jobinfo_share_t **ShareLevel** (void) const
- int **Reruns** (void) const
- unsigned long long int **DiskSpace** (void)
- bool **SwitchUser** (bool su=true) const

4.28.1 Detailed Description

Job run under the privilages of one of the system user. This class collects information related to this user

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

- job_user.h

4.29 ARex::NGConfig Class Reference

```
#include <ngconfig.h>
```

4.29.1 Detailed Description

Configuration class used for reading configuration files ARC-style.

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

- config/ngconfig.h

4.30 ARex::NGConfig Class Reference

```
#include <ngconfig.h>
```

4.30.1 Detailed Description

Configuration class used for reading configuration files ARC-style.

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

- config/ngconfig.h

4.31 ARex::PayloadFile Class Reference

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

4.31.1 Detailed Description

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

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

- a-rex/PayloadFile.h

4.32 Hopi::PayloadFile Class Reference

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

Public Member Functions

- PayloadFile (const char ∗filename)
- PayloadFile (const char ∗filename, int size)
- virtual char **operator[ ]** (int pos) const
- virtual char ∗ **Content** (int pos=-1)
- virtual int **Size** (void) const
- virtual char ∗ **Insert** (int pos=0, int size=0)
- virtual char ∗ **Insert** (const char ∗s, int pos=0, int size=0)
- virtual char ∗ **Buffer** (unsigned int num)
- virtual int **BufferSize** (unsigned int num) const
- virtual int **BufferPos** (unsigned int num) const
- virtual bool **Truncate** (unsigned int size)
- **operator bool** (void)
- bool **operator!** (void)

Protected Attributes

- int **handle_**
- char ∗ **addr_**
- size_t **size_**

4.32.1 Detailed Description

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

4.32.2 Constructor & Destructor Documentation

4.32.2.1 Hopi::PayloadFile::PayloadFile (const char ∗ *filename*)

Creates object associated with file for reading from it

4.32.2.2 Hopi::PayloadFile::PayloadFile (const char ∗ *filename*, int *size*)

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

4.33 ARex::PayloadFile Class Reference

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

4.33.1 Detailed Description

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

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

- a-rex/PayloadFile.h

4.34 RTE Class Reference

```
#include <rte.h>
```

Public Member Functions

- RTE (const std::string &re)
- ∼RTE ()
- std::string str () const
- std::string Name () const
- std::string Version () const
- bool operator== (const RTE &other) const
- bool operator!= (const RTE &other) const
- bool operator> (const RTE &other) const
- bool operator< (const RTE &other) const
- bool operator>= (const RTE &other) const
- bool operator<= (const RTE &other) const

4.34.1 Detailed Description

RTE class. It represents a runtime environment, and provides functionality for getting information about them.

4.34.2 Constructor & Destructor Documentation

4.34.2.1 RTE::RTE (const std::string & *re*)

Constructs a new runtime environemt. String should in general be of the type: STRING-VERSION. Where version consists of numbers with . between them.

4.34.2.2 RTE::∼RTE ()

Destructor. Not that much to say.

4.34.3 Member Function Documentation

4.34.3.1 std::string RTE::Name () const

Returns the name of the runtime environment.

4.34.3.2 bool RTE::operator!= (const RTE & *other*) const

Inequility operator. Return the opsite of ==

4.34.3.3 bool RTE::operator< (const RTE & *other*) const

Less than operator. Returns false if the other is equal, otherwise it returns the opposite of >

4.34.3.4 bool RTE::operator<= (const RTE & *other*) const

Less than or equal operator. Returns the oppsite of >

4.34.3.5 bool RTE::operator== (const RTE & *other*) const

Equliaty operator. Returns true if the runtime environments have the string representation.

4.34.3.6 bool RTE::operator> (const RTE & *other*) const

Greater than operator. Returns true if the compared runtime environment is greater than the current.

4.34.3.7 bool RTE::operator>= (const RTE & *other*) const

Greater or equal operator. Returns the opposite of <

4.34.3.8 std::string RTE::str () const

Returns a string representation of the runtime environment. This is usually the same as given in the constructor.

4.34.3.9 std::string RTE::Version () const

Returns the version of the runtime environment.

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

- rte.h

4.35 RuntimeEnvironment Class Reference

```
#include <runtimeenvironment.h>
```

Public Member Functions

- RuntimeEnvironment (const std::string &re)
- ∼RuntimeEnvironment ()
- std::string str () const
- std::string Name () const
- std::string Version () const
- bool operator== (const RuntimeEnvironment &other) const
- bool operator!= (const RuntimeEnvironment &other) const
- bool operator> (const RuntimeEnvironment &other) const
- bool operator< (const RuntimeEnvironment &other) const
- bool operator>= (const RuntimeEnvironment &other) const
- bool operator<= (const RuntimeEnvironment &other) const

4.35.1 Detailed Description

RuntimeEnvironment class. It represents a runtime environment, and provides functionality for getting information about them.

4.35.2 Constructor & Destructor Documentation

4.35.2.1 RuntimeEnvironment::RuntimeEnvironment (const std::string & *re*)

Constructs a new runtime environemt. String should in general be of the type: STRING-VERSION. Where version consists of numbers with . between them.

4.35.2.2 RuntimeEnvironment::∼RuntimeEnvironment ()

Destructor. Not that much to say.

4.35.3 Member Function Documentation

4.35.3.1 std::string RuntimeEnvironment::Name () const

Returns the name of the runtime environment.

4.35.3.2 bool RuntimeEnvironment::operator!= (const RuntimeEnvironment & *other*) const

Inequility operator. Return the opsite of ==

4.35.3.3 bool RuntimeEnvironment::operator< (const RuntimeEnvironment & *other*) const

Less than operator. Returns false if the other is equal, otherwise it returns the opposite of >

4.35.3.4 bool RuntimeEnvironment::operator<= (const RuntimeEnvironment & *other*) const

Less than or equal operator. Returns the oppsite of >

4.35.3.5 bool RuntimeEnvironment::operator== (const RuntimeEnvironment & *other*) const

Equliaty operator. Returns true if the runtime environments have the string representation.

4.35.3.6 bool RuntimeEnvironment::operator> (const RuntimeEnvironment & *other*) const

Greater than operator. Returns true if the compared runtime environment is greater than the current.

4.35.3.7 bool RuntimeEnvironment::operator>= (const RuntimeEnvironment & *other*) const

Greater or equal operator. Returns the opposite of <

4.35.3.8 std::string RuntimeEnvironment::str () const

Returns a string representation of the runtime environment. This is usually the same as given in the constructor.

4.35.3.9 std::string RuntimeEnvironment::Version () const

Returns the version of the runtime environment.

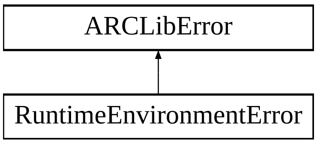
The documentation for this class was generated from the following file:

- runtimeenvironment.h

4.36 RuntimeEnvironmentError Class Reference

```
#include <runtimeenvironment.h>
```

Inheritance diagram for RuntimeEnvironmentError::



Public Member Functions

- RuntimeEnvironmentError (std::string message)

4.36.1 Detailed Description

RuntimeEnvironment exceptions. Gets thrown when an error occurs regarding a runtime environment.

4.36.2 Constructor & Destructor Documentation

4.36.2.1 RuntimeEnvironmentError::RuntimeEnvironmentError (std::string *message*) `[inline]`

Standard exception class constructor.

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

- runtimeenvironment.h

4.37 ArcSec::Service_AA Class Reference

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

Public Member Functions

- **Service_AA** (Arc::Config ∗cfg)
- virtual Arc::MCC_Status **process** (Arc::Message &inmsg, Arc::Message &outmsg)

Protected Member Functions

- Arc::MCC_Status **make_soap_fault** (Arc::Message &outmsg)

Protected Attributes

- Arc::NS **ns_**
- Arc::Logger **logger_**

4.37.1 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

4.38 Compiler::Service_Compiler Class Reference

```
#include <compiler.h>
```

Public Member Functions

- Service_Compiler (Arc::Config ∗cfg)
- virtual Arc::MCC_Status process (Arc::Message &, Arc::Message &)
- std::string **Get_Script_Url** ()

Protected Member Functions

- Arc::MCC_Status **make_fault** (Arc::Message &outmsg)

Protected Attributes

- std::string **script_url_**
- Arc::NS **ns_**

Static Protected Attributes

- static Arc::Logger **logger**

4.38.1 Detailed Description

This service need in the server config a "compiler:scriptfile_url" element. It is the scriptfile place, wherefrom it will be download the JSDL.

4.38.2 Constructor & Destructor Documentation

4.38.2.1 Compiler::Service_Compiler::Service_Compiler (Arc::Config ∗ *cfg*)

Constructor accepts configuration describing content of scriptfile_url

4.38.3 Member Function Documentation

4.38.3.1 virtual Arc::MCC_Status Compiler::Service_Compiler::process (Arc::Message &, Arc::Message &) `[virtual]`

Service request processing routine

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

- compiler.h

4.39 ArcSec::Service_Delegation Class Reference

A Service which launches the proxy certificate request; it accepts the request from.

```
#include <delegation.h>
```

Public Member Functions

- **Service_Delegation** (Arc::Config ∗cfg)
- virtual Arc::MCC_Status **process** (Arc::Message &inmsg, Arc::Message &outmsg)

Protected Member Functions

- Arc::MCC_Status **make_soap_fault** (Arc::Message &outmsg)

Protected Attributes

- Arc::NS **ns_**
- Arc::Logger **logger_**

4.39.1 Detailed Description

A Service which launches the proxy certificate request; it accepts the request from.

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

- delegation.h

4.40 ArcSec::Service_SLCS Class Reference

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

Public Member Functions

- **Service_SLCS** (Arc::Config ∗cfg)
- virtual Arc::MCC_Status **process** (Arc::Message &inmsg, Arc::Message &outmsg)

Protected Member Functions

- Arc::MCC_Status **make_soap_fault** (Arc::Message &outmsg)

Protected Attributes

- Arc::NS **ns_**
- Arc::Logger **logger_**

4.40.1 Detailed Description

A Service which signs the short-lived certificate; it accepts the certificate request 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

4.41 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 &)

Protected Attributes

- Arc::NS **ns_**
- Arc::Logger **logger**
- Arc::XMLNode **metadata_node_**
- std::string **sp_name_**
- std::string **cert_file_**
- std::string **privkey_file_**

4.41.1 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

4.41.2 Constructor & Destructor Documentation

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

Constructor

4.41.3 Member Function Documentation

4.41.3.1 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

4.42 Time Class Reference

```
#include <datetime.h>
```

Public Member Functions

- Time ()
- Time (const time_t &)
- Time (const std::string &)
- Time & operator= (const time_t &)
- void SetTime (const time_t &)
- time_t GetTime () const
- operator std::string () const
- std::string str (const TimeFormat &=time_format) const
- bool operator< (const Time &) const
- bool operator> (const Time &) const
- bool operator<= (const Time &) const
- bool operator>= (const Time &) const
- bool operator== (const Time &) const
- bool operator!= (const Time &) const
- Time ()
- Time (const time_t &)
- Time (const std::string &)
- Time & operator= (const time_t &)
- void SetTime (const time_t &)
- time_t GetTime () const
- operator std::string () const
- std::string str (const TimeFormat &=time_format) const
- bool operator< (const Time &) const
- bool operator> (const Time &) const
- bool operator<= (const Time &) const
- bool operator>= (const Time &) const
- bool operator== (const Time &) const
- bool operator!= (const Time &) const

Static Public Member Functions

- static void SetFormat (const TimeFormat &)
- static TimeFormat GetFormat ()
- static void SetFormat (const TimeFormat &)
- static TimeFormat GetFormat ()

4.42.1 Detailed Description

A class for storing and manipulating times.

4.42.2 Constructor & Destructor Documentation

4.42.2.1 Time::Time ()

Default constructor. The time is put equal the current time.

4.42.2.2 Time::Time (const time_t &)

Constructor that takes a time_t variable and stores it.

4.42.2.3 Time::Time (const std::string &)

Constructor that tries to convert a string into a time_t.

4.42.2.4 Time::Time ()

Default constructor. The time is put equal the current time.

4.42.2.5 Time::Time (const time_t &)

Constructor that takes a time_t variable and stores it.

4.42.2.6 Time::Time (const std::string &)

Constructor that tries to convert a string into a time_t.

4.42.3 Member Function Documentation

4.42.3.1 static TimeFormat Time::GetFormat () `[static]`

Gets the default format for time strings.

4.42.3.2 static TimeFormat Time::GetFormat () `[static]`

Gets the default format for time strings.

4.42.3.3 time_t Time::GetTime () const

gets the time

4.42.3.4 time_t Time::GetTime () const

gets the time

4.42.3.5 Time::operator std::string () const

Returns a string representation of the time, using the default format.

4.42.3.6 Time::operator std::string () const

Returns a string representation of the time, using the default format.

4.42.3.7 bool Time::operator!= (const Time &) const

Comparing two Time objects.

4.42.3.8 bool Time::operator!= (const Time &) const

Comparing two Time objects.

4.42.3.9 bool Time::operator< (const Time &) const

Comparing two Time objects.

4.42.3.10 bool Time::operator< (const Time &) const

Comparing two Time objects.

4.42.3.11 bool Time::operator<= (const Time &) const

Comparing two Time objects.

4.42.3.12 bool Time::operator<= (const Time &) const

Comparing two Time objects.

4.42.3.13 Time& Time::operator= (const time_t &)

Assignment operator from a time_t.

4.42.3.14 Time& Time::operator= (const time_t &)

Assignment operator from a time_t.

4.42.3.15 bool Time::operator== (const Time &) const

Comparing two Time objects.

4.42.3.16 bool Time::operator== (const Time &) const

Comparing two Time objects.

4.42.3.17 bool Time::operator> (const Time &) const

Comparing two Time objects.

4.42.3.18 bool Time::operator> (const Time &) const

Comparing two Time objects.

4.42.3.19 bool Time::operator>= (const Time &) const

Comparing two Time objects.

4.42.3.20 bool Time::operator>= (const Time &) const

Comparing two Time objects.

4.42.3.21 static void Time::SetFormat (const TimeFormat &) `[static]`

Sets the default format for time strings.

4.42.3.22 static void Time::SetFormat (const TimeFormat &) `[static]`

Sets the default format for time strings.

4.42.3.23 void Time::SetTime (const time_t &)

sets the time

4.42.3.24 void Time::SetTime (const time_t &)

sets the time

4.42.3.25 std::string Time::str (const TimeFormat & = `time_format`) const

Returns a string representation of the time, using the specified format.

4.42.3.26 std::string Time::str (const TimeFormat & = `time_format`) const

Returns a string representation of the time, using the specified format.

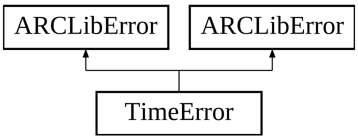
The documentation for this class was generated from the following files:

- grid-manager/jobdesc/datetime.h
- local/grid-manager/jobdesc/datetime.h

4.43 TimeError Class Reference

`#include <datetime.h>`

Inheritance diagram for TimeError::



Public Member Functions

- TimeError (std::string message)
- TimeError (std::string message)

4.43.1 Detailed Description

Class to represent errors thrown by the Time class.

4.43.2 Constructor & Destructor Documentation

4.43.2.1 TimeError::TimeError (std::string *message*) `[inline]`

Standard exception class constructor.

4.43.2.2 TimeError::TimeError (std::string *message*) `[inline]`

Standard exception class constructor.

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

- grid-manager/jobdesc/datetime.h
- local/grid-manager/jobdesc/datetime.h

4.44 ARex::XMLConfig Class Reference

```
#include <xmlconfig.h>
```

4.44.1 Detailed Description

Class for reading in configuration files in xml-format. It uses libxml2 for xml-parsing.

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

- config/xmlconfig.h

4.45 ARex::XMLConfig Class Reference

```
#include <xmlconfig.h>
```

4.45.1 Detailed Description

Class for reading in configuration files in xml-format. It uses libxml2 for xml-parsing.

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

- config/xmlconfig.h

4.46 Xrsl Class Reference

`#include <xrsl.h>`

Public Member Functions

- Xrsl (const std::string &xrsl_string) throw (XrslError)
- Xrsl (xrsl_operator=operator_and)
- Xrsl (globus_rsl_t ∗)
- Xrsl (const Xrsl &other_xrsl)
- Xrsl & operator= (const Xrsl &other_xrsl)
- ∼Xrsl ()
- void Print () const
- const std::string str () const throw (XrslError)
- std::list< Xrsl > SplitMulti ()
- std::list< Xrsl > SplitOrRelation () throw (XrslError)
- void AddRelation (const XrslRelation &relation, bool force=true) throw (XrslError)
- void AddSimpleRelation (const std::string &attr, xrsl_operator op, const std::string &val, bool force=true) throw (XrslError)
- void AddXrsl (Xrsl &axrsl) throw (XrslError)
- XrslRelation GetRelation (const std::string &attr) throw (XrslError)
- std::list< XrslRelation > GetAllRelations (const std::string &attr)
- bool IsRelation (const std::string &)
- void RemoveRelation (const std::string &attr) throw (XrslError)
- void Validate (const std::list< XrslValidationData > &valid_attributes, bool allow_unknown=false) throw (XrslError)
- void Eval ()

- Xrsl (const std::string &xrsl_string) throw (XrslError)
- Xrsl (xrsl_operator=operator_and)
- Xrsl (globus_rsl_t ∗)
- Xrsl (const Xrsl &other_xrsl)
- Xrsl & operator= (const Xrsl &other_xrsl)
- ∼Xrsl ()
- void Print () const
- const std::string str () const throw (XrslError)
- std::list< Xrsl > SplitMulti ()
- std::list< Xrsl > SplitOrRelation () throw (XrslError)
- void AddRelation (const XrslRelation &relation, bool force=true) throw (XrslError)
- void AddSimpleRelation (const std::string &attr, xrsl_operator op, const std::string &val, bool force=true) throw (XrslError)
- void AddXrsl (Xrsl &axrsl) throw (XrslError)
- XrslRelation GetRelation (const std::string &attr) throw (XrslError)
- std::list< XrslRelation > GetAllRelations (const std::string &attr)
- bool IsRelation (const std::string &)
- void RemoveRelation (const std::string &attr) throw (XrslError)
- void Validate (const std::list< XrslValidationData > &valid_attributes, bool allow_unknown=false) throw (XrslError)
- void Eval ()

4.46.1 Detailed Description

Class used to simplify manipulation of xRSL job descriptions.

4.46.2 Constructor & Destructor Documentation

4.46.2.1 Xrsl::Xrsl (const std::string & *xrsl_string*) throw (XrslError)

Constructs a Xrsl object from a string respresentation.

4.46.2.2 Xrsl::Xrsl (xrsl_operator = `operator_and`)

Constructs empty Xrsl object.

4.46.2.3 Xrsl::Xrsl (globus_rsl_t ∗)

Construct Xrsl object from globus_rsl_t∗ .

4.46.2.4 Xrsl::Xrsl (const Xrsl & *other_xrsl*)

Copy constructor.

4.46.2.5 Xrsl::∼Xrsl ()

Destructor.

4.46.2.6 Xrsl::Xrsl (const std::string & *xrsl_string*) throw (XrslError)

Constructs a Xrsl object from a string respresentation.

4.46.2.7 Xrsl::Xrsl (xrsl_operator = `operator_and`)

Constructs empty Xrsl object.

4.46.2.8 Xrsl::Xrsl (globus_rsl_t ∗)

Construct Xrsl object from globus_rsl_t∗ .

4.46.2.9 Xrsl::Xrsl (const Xrsl & *other_xrsl*)

Copy constructor.

4.46.2.10 Xrsl::∼Xrsl ()

Destructor.

4.46.3 Member Function Documentation

4.46.3.1 void Xrsl::AddRelation (const XrslRelation & *relation*, bool *force* = `true`) throw (XrslError)

Adds a new relation. Throws exception if relation already exists in the xrsl and force is not true.

4.46.3.2 void Xrsl::AddRelation (const XrslRelation & *relation*, bool *force* = `true`) throw (XrslError)

Adds a new relation. Throws exception if relation already exists in the xrsl and force is not true.

4.46.3.3 void Xrsl::AddSimpleRelation (const std::string & *attr*, xrsl_operator *op*, const std::string & *val*, bool *force* = `true`) throw (XrslError)

Adds simple relation specified by attribute, xrsl-operator and value. Throws exception if relation already exists in the xrsl and force is not true.

4.46.3.4 void Xrsl::AddSimpleRelation (const std::string & *attr*, xrsl_operator *op*, const std::string & *val*, bool *force* = `true`) throw (XrslError)

Adds simple relation specified by attribute, xrsl-operator and value. Throws exception if relation already exists in the xrsl and force is not true.

4.46.3.5 void Xrsl::AddXrsl (Xrsl & *axrsl*) throw (XrslError)

Adds a sub-Xrsl to the Xrsl.

4.46.3.6 void Xrsl::AddXrsl (Xrsl & *axrsl*) throw (XrslError)

Adds a sub-Xrsl to the Xrsl.

4.46.3.7 void Xrsl::Eval ()

Performs RSL alias substitution etc.

4.46.3.8 void Xrsl::Eval ()

Performs RSL alias substitution etc.

4.46.3.9 std::list<XrslRelation> Xrsl::GetAllRelations (const std::string & *attr*)

Get all XrslRelation’s in the xrsl with attribute equal to parameter attr.

4.46.3.10 std::list<XrslRelation> Xrsl::GetAllRelations (const std::string & *attr*)

Get all XrslRelation’s in the xrsl with attribute equal to parameter attr.

4.46.3.11 XrslRelation Xrsl::GetRelation (const std::string & *attr*) throw (XrslError)

Gets the first XrslRelation corresponding to the attribute.

4.46.3.12 XrslRelation Xrsl::GetRelation (const std::string & *attr*) throw (XrslError)

Gets the first XrslRelation corresponding to the attribute.

4.46.3.13 bool Xrsl::IsRelation (const std::string &)

Does the relation with this attribute exist?

4.46.3.14 bool Xrsl::IsRelation (const std::string &)

Does the relation with this attribute exist?

4.46.3.15 Xrsl& Xrsl::operator= (const Xrsl & *other_xrsl*)

Copy-assignment constructor.

4.46.3.16 Xrsl& Xrsl::operator= (const Xrsl & *other_xrsl*)

Copy-assignment constructor.

4.46.3.17 void Xrsl::Print () const

Print detailed information about each relation.

4.46.3.18 void Xrsl::Print () const

Print detailed information about each relation.

4.46.3.19 void Xrsl::RemoveRelation (const std::string & *attr*) throw (XrslError)

Removes a relation. Throws an exception if the relation does not exist in the xrsl. The relation may be of any type.

4.46.3.20 void Xrsl::RemoveRelation (const std::string & *attr*) throw (XrslError)

Removes a relation. Throws an exception if the relation does not exist in the xrsl. The relation may be of any type.

4.46.3.21 std::list<Xrsl> Xrsl::SplitMulti ()

If the Xrsl start with a +, split the Xrsl into multiple Xrsls.

4.46.3.22 std::list<Xrsl> Xrsl::SplitMulti ()

If the Xrsl start with a +, split the Xrsl into multiple Xrsls.

4.46.3.23 std::list<Xrsl> Xrsl::SplitOrRelation () throw (XrslError)

Splits an Xrsl containing or-operators into separate Xrsl's. Example: &(executable=/bin/echo)(|(cluster=cl1)(cluster=cl2)) split into &(executable=/bin/echo)(cluster=cl1) and &(executable=/bin/echo)(cluster=cl2).

4.46.3.24 std::list<Xrsl> Xrsl::SplitOrRelation () throw (XrslError)

Splits an Xrsl containing or-operators into separate Xrsl's. Example: &(executable=/bin/echo)(|(cluster=cl1)(cluster=cl2)) split into &(executable=/bin/echo)(cluster=cl1) and &(executable=/bin/echo)(cluster=cl2).

4.46.3.25 const std::string Xrsl::str () const throw (XrslError)

Converts the Xrsl object to std::string representation.

4.46.3.26 const std::string Xrsl::str () const throw (XrslError)

Converts the Xrsl object to std::string representation.

4.46.3.27 void Xrsl::Validate (const std::list< XrslValidationData > & *valid_attributes*, bool *allow_unknown* = `false`) throw (XrslError)

Ensures that the xrsl only contains valid attributes. Throws exception if some attribute has invalid format, or that some mandatory attribute is missing.

4.46.3.28 void Xrsl::Validate (const std::list< XrslValidationData > & *valid_attributes*, bool *allow_unknown* = `false`) throw (XrslError)

Ensures that the xrsl only contains valid attributes. Throws exception if some attribute has invalid format, or that some mandatory attribute is missing.

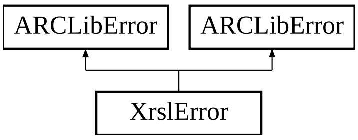
The documentation for this class was generated from the following files:

- grid-manager/jobdesc/xrsl.h
- local/grid-manager/jobdesc/xrsl.h

4.47 XrslError Class Reference

```
#include <xrsl.h>
```

Inheritance diagram for XrslError::



Public Member Functions

- XrslError (std::string message_arg)
- XrslError (std::string message_arg)

4.47.1 Detailed Description

Class represents exceptions associated with usage of the Xrsl class.

4.47.2 Constructor & Destructor Documentation

4.47.2.1 XrslError::XrslError (std::string *message_arg*) `[inline]`

Standard exception constructor.

4.47.2.2 XrslError::XrslError (std::string *message_arg*) `[inline]`

Standard exception constructor.

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

- grid-manager/jobdesc/xrsl.h
- local/grid-manager/jobdesc/xrsl.h

4.48 XrslRelation Class Reference

```
#include <xrsl.h>
```

Public Member Functions

- XrslRelation (const std::string &attribute, const xrsl_operator &oper, const std::string &value)
- XrslRelation (const std::string &attribute, const xrsl_operator &oper, const std::list< std::string > &value)
- XrslRelation (const std::string &attribute, const xrsl_operator &oper, const std::list< std::list< std::string > > &value)
- XrslRelation (globus_rsl_t ∗relation)
- XrslRelation (const XrslRelation &other_relation)
- ∼XrslRelation ()
- std::string str ()
- std::string GetAttribute () const
- xrsl_operator GetOperator () const
- std::string GetSingleValue () throw (XrslError)
- std::list< std::string > GetListValue () throw (XrslError)
- std::list< std::list< std::string > > GetDoubleListValue () throw (XrslError)
- globus_rsl_t ∗ GetRelation () const
- XrslRelation (const std::string &attribute, const xrsl_operator &oper, const std::string &value)
- XrslRelation (const std::string &attribute, const xrsl_operator &oper, const std::list< std::string > &value)
- XrslRelation (const std::string &attribute, const xrsl_operator &oper, const std::list< std::list< std::string > > &value)
- XrslRelation (globus_rsl_t ∗relation)
- XrslRelation (const XrslRelation &other_relation)
- ∼XrslRelation ()
- std::string str ()
- std::string GetAttribute () const
- xrsl_operator GetOperator () const
- std::string GetSingleValue () throw (XrslError)
- std::list< std::string > GetListValue () throw (XrslError)
- std::list< std::list< std::string > > GetDoubleListValue () throw (XrslError)
- globus_rsl_t ∗ GetRelation () const

4.48.1 Detailed Description

XrslRelation class that describes an Xrsl-relation with an attribute, an operator and a value. Various constructors and methods for extracting the attribute and the value (single value or list) are given.

4.48.2 Constructor & Destructor Documentation

4.48.2.1 XrslRelation::XrslRelation (const std::string & *attribute*, const xrsl_operator & *oper*, const std::string & *value*)

Constructor constructing an xrsl-relation from an attribute, an operator and a value.

4.48.2.2 XrslRelation::XrslRelation (const std::string & *attribute*, const xrsl_operator & *oper*, const std::list< std::string > & *value*)

Constructor constructing an xrsl-relation from an attribute, an operator and a value-list.

4.48.2.3 XrslRelation::XrslRelation (const std::string & *attribute*, const xrsl_operator & *oper*, const std::list< std::list< std::string > > & *value*)

Constructor constructing an xrsl-relation from an attribute, an operator and a double value-list.

4.48.2.4 XrslRelation::XrslRelation (globus_rsl_t ∗ *relation*)

Constructs a relation from a globus_rsl_t∗.

4.48.2.5 XrslRelation::XrslRelation (const XrslRelation & *other_relation*)

Copy-constructor.

4.48.2.6 XrslRelation::∼XrslRelation ()

Destructor.

4.48.2.7 XrslRelation::XrslRelation (const std::string & *attribute*, const xrsl_operator & *oper*, const std::string & *value*)

Constructor constructing an xrsl-relation from an attribute, an operator and a value.

4.48.2.8 XrslRelation::XrslRelation (const std::string & *attribute*, const xrsl_operator & *oper*, const std::list< std::string > & *value*)

Constructor constructing an xrsl-relation from an attribute, an operator and a value-list.

4.48.2.9 XrslRelation::XrslRelation (const std::string & *attribute*, const xrsl_operator & *oper*, const std::list< std::list< std::string > > & *value*)

Constructor constructing an xrsl-relation from an attribute, an operator and a double value-list.

4.48.2.10 XrslRelation::XrslRelation (globus_rsl_t ∗ *relation*)

Constructs a relation from a globus_rsl_t∗.

4.48.2.11 XrslRelation::XrslRelation (const XrslRelation & *other_relation*)

Copy-constructor.

4.48.2.12 XrslRelation::∼XrslRelation ()

Destructor.

4.48.3 Member Function Documentation

4.48.3.1 std::string XrslRelation::GetAttribute () const

Returns the attribute of the relation.

4.48.3.2 std::string XrslRelation::GetAttribute () const

Returns the attribute of the relation.

4.48.3.3 std::list<std::list<std::string> > XrslRelation::GetDoubleListValue () throw (XrslError)

If the value of the relation is a double list value, return it.

4.48.3.4 std::list<std::list<std::string> > XrslRelation::GetDoubleListValue () throw (XrslError)

If the value of the relation is a double list value, return it.

4.48.3.5 std::list<std::string> XrslRelation::GetListValue () throw (XrslError)

If the value of the relation is a list value, return it.

4.48.3.6 std::list<std::string> XrslRelation::GetListValue () throw (XrslError)

If the value of the relation is a list value, return it.

4.48.3.7 xrsl_operator XrslRelation::GetOperator () const

Returns the xrsl_operator of the attribute.

4.48.3.8 xrsl_operator XrslRelation::GetOperator () const

Returns the xrsl_operator of the attribute.

4.48.3.9 globus_rsl_t∗ XrslRelation::GetRelation () const

Returns relation.

4.48.3.10 globus_rsl_t∗ XrslRelation::GetRelation () const

Returns relation.

4.48.3.11 std::string XrslRelation::GetSingleValue () throw (XrslError)

If the value of the representation is a single string value, return it.

4.48.3.12 std::string XrslRelation::GetSingleValue () throw (XrslError)

If the value of the representation is a single string value, return it.

4.48.3.13 std::string XrslRelation::str ()

Returns a strng representation of the relation.

4.48.3.14 std::string XrslRelation::str ()

Returns a strng representation of the relation.

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

- grid-manager/jobdesc/xrsl.h
- local/grid-manager/jobdesc/xrsl.h

4.49 XrslValidationData Class Reference

```
#include <xrsl.h>
```

Public Member Functions

- XrslValidationData (const std::string &attribute_name, relation_type rel_type, validation_type val_type, bool unique=true, int list_length=0)
- XrslValidationData (const std::string &attribute_name, relation_type rel_type, validation_type val_type, bool unique=true, int list_length=0)

Public Attributes

- std::string attribute_name
- relation_type rel_type
- bool unique
- int list_length
- validation_type val_type

4.49.1 Detailed Description

Class for simplifying Xrsl validation. One object of this class represents a valid attribute in the xrsl.

4.49.2 Constructor & Destructor Documentation

4.49.2.1 XrslValidationData::XrslValidationData (const std::string & *attribute_name*, relation_type *rel_type*, validation_type *val_type*, bool *unique* = `true`, int *list_length* = `0`)

Constructor.

4.49.2.2 XrslValidationData::XrslValidationData (const std::string & *attribute_name*, relation_type *rel_type*, validation_type *val_type*, bool *unique* = `true`, int *list_length* = `0`)

Constructor.

4.49.3 Member Data Documentation

4.49.3.1 std::string XrslValidationData::attribute_name

Name of attribute.

4.49.3.2 int XrslValidationData::list_length

Length of each list in case attribute is a list of values.

4.49.3.3 relation_type XrslValidationData::rel_type

Type the attribute must have.

4.49.3.4 bool XrslValidationData::unique

Must this attribute be unique?

4.49.3.5 validation_type XrslValidationData::val_type

Must the Xrsl must contain this attribute to be valid?

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

- grid-manager/jobdesc/xrsl.h
- local/grid-manager/jobdesc/xrsl.h

Chapter 5

Hosting Environment (Daemon) Services File Documentation

5.1 configcore.h File Reference

```
#include <iostream>
#include <list>
#include <map>
#include <string>
#include <arc/Logger.h>
```

Namespaces

- namespace **ARex**

Classes

- class ARex::ConfigError
- class **ARex::Option**
- class **ARex::ConfGrp**
- class ARex::Config

Functions

- Config ARex::ReadConfig (std::istream &is)
- Config ARex::ReadConfig (const std::string &filename)

Variables

- Arc::Logger **ARex::ConfigLogger**

5.1.1 Detailed Description

This file describes the core configuration

5.2 configcore.h File Reference

```
#include <iostream>
#include <list>
#include <map>
#include <string>
#include <arc/Logger.h>
```

Namespaces

- namespace **ARex**

Classes

- class ARex::ConfigError
- class **ARex::Option**
- class **ARex::ConfGrp**
- class ARex::Config

Functions

- Config ARex::ReadConfig (std::istream &is)
- Config ARex::ReadConfig (const std::string &filename)

Variables

- Arc::Logger **ARex::ConfigLogger**

5.2.1 Detailed Description

This file describes the core configuration

Index