

## Hosting Environment (Daemon) Services

Generated by Doxygen 1.6.1

Sat Feb 2 01:01:19 2013



# Contents

<b>1</b>	<b>Data Structure Index</b>	<b>1</b>
1.1	Class Hierarchy . . . . .	1
<b>2</b>	<b>Data Structure Index</b>	<b>5</b>
2.1	Data Structures . . . . .	5
<b>3</b>	<b>Data Structure Documentation</b>	<b>9</b>
3.1	Arc::ApelDestination Class Reference . . . . .	9
3.1.1	Detailed Description . . . . .	9
3.1.2	Constructor & Destructor Documentation . . . . .	9
3.1.2.1	ApelDestination . . . . .	9
3.1.3	Member Function Documentation . . . . .	9
3.1.3.1	finish . . . . .	9
3.1.3.2	report . . . . .	10
3.2	ARex::ARexGMConfig Class Reference . . . . .	11
3.3	ARex::ARexJob Class Reference . . . . .	12
3.3.1	Detailed Description . . . . .	12
3.3.2	Constructor & Destructor Documentation . . . . .	12
3.3.2.1	ARexJob . . . . .	12
3.3.2.2	ARexJob . . . . .	13
3.3.3	Member Function Documentation . . . . .	13
3.3.3.1	Cancel . . . . .	13
3.3.3.2	ChooseSessionDir . . . . .	13
3.3.3.3	Clean . . . . .	13
3.3.3.4	Created . . . . .	13
3.3.3.5	CreateFile . . . . .	13
3.3.3.6	Failed . . . . .	13
3.3.3.7	FailedState . . . . .	13
3.3.3.8	Failure . . . . .	13

3.3.3.9	GetDescription	13
3.3.3.10	ID	14
3.3.3.11	Jobs	14
3.3.3.12	LogDir	14
3.3.3.13	LogFiles	14
3.3.3.14	Modified	14
3.3.3.15	OpenDir	14
3.3.3.16	OpenFile	14
3.3.3.17	OpenLogFile	14
3.3.3.18	Resume	14
3.3.3.19	SessionDir	14
3.3.3.20	State	14
3.3.3.21	State	15
3.3.3.22	TotalJobs	15
3.3.3.23	UpdateCredentials	15
3.4	ARex::ARexService Class Reference	16
3.5	AuthEvaluator Class Reference	17
3.6	AuthUser Class Reference	18
3.7	AuthVO Class Reference	19
3.8	ARex::CacheConfig Class Reference	20
3.8.1	Detailed Description	20
3.8.2	Constructor & Destructor Documentation	20
3.8.2.1	CacheConfig	20
3.8.2.2	CacheConfig	20
3.8.2.3	CacheConfig	20
3.9	ARex::CacheConfigException Class Reference	21
3.9.1	Detailed Description	21
3.10	Cache::CacheService Class Reference	22
3.10.1	Detailed Description	22
3.10.2	Constructor & Destructor Documentation	22
3.10.2.1	CacheService	22
3.10.2.2	~CacheService	22
3.10.3	Member Function Documentation	22
3.10.3.1	CacheCheck	22
3.10.3.2	CacheLink	23
3.10.3.3	CacheLinkQuery	23

3.10.3.4	operator bool . . . . .	23
3.10.3.5	operator! . . . . .	23
3.10.3.6	process . . . . .	23
3.10.3.7	RegistrationCollector . . . . .	23
3.11	Cache::CacheServiceGenerator Class Reference . . . . .	24
3.11.1	Detailed Description . . . . .	24
3.11.2	Constructor & Destructor Documentation . . . . .	24
3.11.2.1	CacheServiceGenerator . . . . .	24
3.11.3	Member Function Documentation . . . . .	24
3.11.3.1	addNewRequest . . . . .	24
3.11.3.2	queryRequestsFinished . . . . .	25
3.12	Arc::CARDestination Class Reference . . . . .	26
3.12.1	Detailed Description . . . . .	26
3.12.2	Constructor & Destructor Documentation . . . . .	26
3.12.2.1	CARDestination . . . . .	26
3.12.3	Member Function Documentation . . . . .	26
3.12.3.1	finish . . . . .	26
3.12.3.2	report . . . . .	26
3.13	ARex::CommFIFO Class Reference . . . . .	27
3.14	gridftpdd::ConfigSections Class Reference . . . . .	28
3.15	ARex::ConfigSections Class Reference . . . . .	29
3.16	ARex::ContinuationPlugins Class Reference . . . . .	30
3.17	ARex::CoreConfig Class Reference . . . . .	31
3.17.1	Detailed Description . . . . .	31
3.18	ARex::CountedResource Class Reference . . . . .	32
3.19	gridftpdd::Daemon Class Reference . . . . .	33
3.20	DataStaging::DataDeliveryService Class Reference . . . . .	34
3.20.1	Detailed Description . . . . .	34
3.21	ARex::DelegationStore Class Reference . . . . .	35
3.22	ARex::DelegationStores Class Reference . . . . .	36
3.22.1	Detailed Description . . . . .	36
3.22.2	Member Function Documentation . . . . .	36
3.22.2.1	DelegatedToken . . . . .	36
3.22.2.2	Process . . . . .	36
3.23	Arc::Destination Class Reference . . . . .	37
3.23.1	Detailed Description . . . . .	37

3.23.2	Member Function Documentation	37
3.23.2.1	createDestination	37
3.23.2.2	finish	37
3.23.2.3	report	37
3.24	Arc::Destinations Class Reference	38
3.24.1	Detailed Description	38
3.24.2	Member Function Documentation	38
3.24.2.1	report	38
3.25	DirectAccess::diraccess_t Struct Reference	39
3.26	DirectAccess Class Reference	40
3.27	DirectFilePlugin Class Reference	41
3.28	DirEntry Class Reference	42
3.29	ARex::DTRGenerator Class Reference	43
3.29.1	Detailed Description	43
3.29.2	Constructor & Destructor Documentation	43
3.29.2.1	DTRGenerator	43
3.29.2.2	~DTRGenerator	43
3.29.3	Member Function Documentation	43
3.29.3.1	cancelJob	43
3.29.3.2	checkUploadedFiles	44
3.29.3.3	hasJob	44
3.29.3.4	queryJobFinished	44
3.29.3.5	receiveDTR	44
3.29.3.6	receiveJob	45
3.29.3.7	removeJob	45
3.30	ARex::DTRInfo Class Reference	46
3.30.1	Detailed Description	46
3.31	Entry Class Reference	47
3.32	ARex::Exec Class Reference	48
3.33	ARex::FileChunks Class Reference	49
3.33.1	Detailed Description	49
3.33.2	Member Function Documentation	49
3.33.2.1	Release	49
3.33.2.2	Remove	49
3.34	ARex::FileChunksList Class Reference	50
3.34.1	Detailed Description	50

3.34.2	Member Function Documentation	50
3.34.2.1	Get	50
3.35	ARex::FileChunksRef Class Reference	51
3.36	ARex::FileData Class Reference	52
3.37	FileNode Class Reference	53
3.38	FilePlugin Class Reference	54
3.39	ARex::FileRecord Class Reference	55
3.40	FileRoot Class Reference	56
3.41	GACLPlugin Class Reference	57
3.42	gm_dirs_ Struct Reference	58
3.43	ARex::GMConfig Class Reference	59
3.43.1	Detailed Description	60
3.43.2	Constructor & Destructor Documentation	61
3.43.2.1	GMConfig	61
3.43.3	Member Function Documentation	61
3.43.3.1	CreateControlDirectory	61
3.43.3.2	CreateSessionDirectory	61
3.43.3.3	Load	61
3.43.3.4	SessionRoot	61
3.43.3.5	Substitute	61
3.44	gridftp::GMEEnvironment Class Reference	62
3.44.1	Member Function Documentation	62
3.44.1.1	nordugrid_config_loc	62
3.44.1.2	support_mail_address	62
3.45	ARex::GMJob Class Reference	63
3.45.1	Detailed Description	63
3.46	GridFTP_Commands Class Reference	64
3.47	GridFTP_Commands_timeout Class Reference	65
3.48	ARex::GridManager Class Reference	66
3.49	Identity Class Reference	67
3.50	IdentityGACL Class Reference	68
3.51	IdentityItemDN Class Reference	69
3.52	IdentityItemVOMS Class Reference	70
3.53	Index Class Reference	71
3.54	ObjectAccess::Item Class Reference	72
3.55	Identity::Item Class Reference	73

3.56	ARex::FileRecord::Iterator Class Reference	74
3.57	ARex::job_state_rec_t Struct Reference	75
3.58	ARex::JobDescriptionHandler Class Reference	76
3.58.1	Detailed Description	76
3.58.2	Member Function Documentation	76
3.58.2.1	parse_job_req	76
3.58.2.2	parse_job_req	76
3.58.2.3	parse_job_req	77
3.59	ARex::JobIDGenerator Class Reference	78
3.60	ARex::JobIDGeneratorARC Class Reference	79
3.61	ARex::JobIDGeneratorES Class Reference	80
3.62	ARex::JobLocalDescription Class Reference	81
3.63	ARex::JobLog Class Reference	82
3.63.1	Detailed Description	82
3.64	Arc::JobLogFile Class Reference	83
3.64.1	Detailed Description	83
3.64.2	Constructor & Destructor Documentation	83
3.64.2.1	JobLogFile	83
3.64.3	Member Function Documentation	83
3.64.3.1	allowRemove	83
3.64.3.2	createCARUsageRecord	83
3.64.3.3	createUsageRecord	83
3.64.3.4	exists	84
3.64.3.5	getFilename	84
3.64.3.6	olderThan	84
3.64.3.7	parse	84
3.64.3.8	remove	84
3.65	JobPlugin Class Reference	85
3.66	ARex::JobReqResult Class Reference	86
3.66.1	Detailed Description	86
3.67	ARex::JobsList Class Reference	87
3.67.1	Detailed Description	87
3.68	gridftp::LdapQuery Class Reference	88
3.68.1	Detailed Description	88
3.68.2	Member Enumeration Documentation	88
3.68.2.1	Scope	88



3.68.3	Constructor & Destructor Documentation	88
3.68.3.1	LdapQuery	88
3.68.3.2	~LdapQuery	88
3.68.4	Member Function Documentation	88
3.68.4.1	Host	88
3.68.4.2	Query	89
3.68.4.3	Result	89
3.69	gridftp::LdapQueryError Class Reference	90
3.69.1	Detailed Description	90
3.69.2	Constructor & Destructor Documentation	90
3.69.2.1	LdapQueryError	90
3.70	ARex::RunPlugin::lib_plugin_t Union Reference	91
3.71	gridftp::RunPlugin::lib_plugin_t Union Reference	92
3.72	ARex::LRMSResult Class Reference	93
3.73	Arc::LutsDestination Class Reference	94
3.73.1	Detailed Description	94
3.73.2	Constructor & Destructor Documentation	94
3.73.2.1	LutsDestination	94
3.73.3	Member Function Documentation	94
3.73.3.1	finish	94
3.73.3.2	report	94
3.74	ObjectAccess Class Reference	95
3.75	ObjectAccessGACL Class Reference	96
3.76	ARex::OptimizedInformationContainer Class Reference	97
3.77	gridftp::ParallelLdapQueries Class Reference	98
3.77.1	Detailed Description	98
3.78	ARex::PayloadBigFile Class Reference	99
3.78.1	Constructor & Destructor Documentation	99
3.78.1.1	PayloadBigFile	99
3.78.1.2	~PayloadBigFile	99
3.79	ARex::PayloadFAFile Class Reference	100
3.79.1	Constructor & Destructor Documentation	100
3.79.1.1	PayloadFAFile	100
3.80	ARex::PayloadFile Class Reference	101
3.80.1	Detailed Description	101
3.80.2	Constructor & Destructor Documentation	101

3.80.2.1	PayloadFile	101
3.80.2.2	~PayloadFile	101
3.81	Permission Class Reference	102
3.82	PermissionGACL Class Reference	103
3.83	Policy Class Reference	104
3.84	ARex::ContinuationPlugins::result_t Class Reference	105
3.85	ARex::RunParallel Class Reference	106
3.85.1	Detailed Description	106
3.86	ARex::RunPlugin Class Reference	107
3.86.1	Detailed Description	107
3.87	gridftpd::RunPlugin Class Reference	108
3.88	ARex::RunRedirected Class Reference	109
3.88.1	Detailed Description	109
3.89	Server Class Reference	110
3.90	FileRoot::ServerParams Class Reference	111
3.91	ArcSec::Service_AA Class Reference	112
3.91.1	Detailed Description	112
3.92	Arc::Service_JavaWrapper Class Reference	113
3.92.1	Member Function Documentation	113
3.92.1.1	process	113
3.93	Arc::Service_PythonWrapper Class Reference	114
3.93.1	Member Function Documentation	114
3.93.1.1	process	114
3.94	ArcSec::Service_SLCS Class Reference	115
3.94.1	Detailed Description	115
3.95	SPService::Service_SP Class Reference	116
3.95.1	Detailed Description	116
3.95.2	Constructor & Destructor Documentation	116
3.95.2.1	Service_SP	116
3.95.3	Member Function Documentation	116
3.95.3.1	process	116
3.96	SimpleMap Class Reference	117
3.97	ARex::StagingConfig Class Reference	118
3.97.1	Detailed Description	118
3.97.2	Constructor & Destructor Documentation	118
3.97.2.1	StagingConfig	118

3.98 UnixMap Class Reference . . . . .	119
3.99 ARex::UrlMapConfig Class Reference . . . . .	120
3.100gridftpd::UrlMapConfig Class Reference . . . . .	121
3.101Arc::UsageReporter Class Reference . . . . .	122
3.101.1 Detailed Description . . . . .	122
3.101.2 Constructor & Destructor Documentation . . . . .	122
3.101.2.1 UsageReporter . . . . .	122
3.101.3 Member Function Documentation . . . . .	122
3.101.3.1 report . . . . .	122
3.102userspec_t Class Reference . . . . .	123
3.103voms Struct Reference . . . . .	124
3.103.1 Detailed Description . . . . .	124
3.103.2 Field Documentation . . . . .	124
3.103.2.1 attrs . . . . .	124
3.103.2.2 server . . . . .	124
3.103.2.3 voname . . . . .	124
3.104voms_attrs Struct Reference . . . . .	125
3.104.1 Detailed Description . . . . .	125
3.104.2 Field Documentation . . . . .	125
3.104.2.1 cap . . . . .	125
3.104.2.2 group . . . . .	125
3.104.2.3 role . . . . .	125
3.105ARex::ZeroUInt Class Reference . . . . .	126
3.105.1 Detailed Description . . . . .	126



# Chapter 1

## Data Structure Index

### 1.1 Class Hierarchy

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

ARex::ARexGMConfig . . . . .	11
ARex::ARexJob . . . . .	12
ARex::ARexService . . . . .	16
AuthEvaluator . . . . .	17
AuthUser . . . . .	18
AuthVO . . . . .	19
ARex::CacheConfig . . . . .	20
ARex::CacheConfigException . . . . .	21
Cache::CacheService . . . . .	22
Cache::CacheServiceGenerator . . . . .	24
ARex::CommFIFO . . . . .	27
gridftp::ConfigSections . . . . .	28
ARex::ConfigSections . . . . .	29
ARex::ContinuationPlugins . . . . .	30
ARex::CoreConfig . . . . .	31
ARex::CountedResource . . . . .	32
gridftp::Daemon . . . . .	33
DataStaging::DataDeliveryService . . . . .	34
ARex::DelegationStore . . . . .	35
ARex::DelegationStores . . . . .	36
Arc::Destination . . . . .	37
Arc::ApelDestination . . . . .	9
Arc::CARDestination . . . . .	26
Arc::LutsDestination . . . . .	94
Arc::Destinations . . . . .	38
DirectAccess::diraccess_t . . . . .	39
DirectAccess . . . . .	40
DirEntry . . . . .	42
ARex::DTRGenerator . . . . .	43
ARex::DTRInfo . . . . .	46
Entry . . . . .	47
ARex::Exec . . . . .	48
ARex::FileChunks . . . . .	49

ARex::FileChunksList . . . . .	50
ARex::FileChunksRef . . . . .	51
ARex::FileData . . . . .	52
FileNode . . . . .	53
FilePlugin . . . . .	54
DirectFilePlugin . . . . .	41
GACLPlugin . . . . .	57
JobPlugin . . . . .	85
ARex::FileRecord . . . . .	55
FileRoot . . . . .	56
gm_dirs_ . . . . .	58
ARex::GMConfig . . . . .	59
gridftpd::GMEEnvironment . . . . .	62
ARex::GMJob . . . . .	63
GridFTP_Commands . . . . .	64
GridFTP_Commands_timeout . . . . .	65
ARex::GridManager . . . . .	66
Identity . . . . .	67
IdentityGACL . . . . .	68
Index . . . . .	71
Identity::Item . . . . .	73
IdentityItemDN . . . . .	69
IdentityItemVOMS . . . . .	70
ObjectAccess::Item . . . . .	72
ARex::FileRecord::Iterator . . . . .	74
ARex::job_state_rec_t . . . . .	75
ARex::JobDescriptionHandler . . . . .	76
ARex::JobIDGenerator . . . . .	78
ARex::JobIDGeneratorARC . . . . .	79
ARex::JobIDGeneratorES . . . . .	80
ARex::JobLocalDescription . . . . .	81
ARex::JobLog . . . . .	82
Arc::JobLogFile . . . . .	83
ARex::JobReqResult . . . . .	86
ARex::JobsList . . . . .	87
gridftpd::LdapQuery . . . . .	88
gridftpd::LdapQueryError . . . . .	90
ARex::RunPlugin::lib_plugin_t . . . . .	91
gridftpd::RunPlugin::lib_plugin_t . . . . .	92
ARex::LRMSResult . . . . .	93
ObjectAccess . . . . .	95
ObjectAccessGACL . . . . .	96
ARex::OptimizedInformationContainer . . . . .	97
gridftpd::ParallelLdapQueries . . . . .	98
ARex::PayloadBigFile . . . . .	99
ARex::PayloadFAFile . . . . .	100
ARex::PayloadFile . . . . .	101
Permission . . . . .	102
PermissionGACL . . . . .	103
Policy . . . . .	104
ARex::ContinuationPlugins::result_t . . . . .	105
ARex::RunParallel . . . . .	106

ARex::RunPlugin . . . . .	107
gridftpd::RunPlugin . . . . .	108
ARex::RunRedirected . . . . .	109
Server . . . . .	110
FileRoot::ServerParams . . . . .	111
ArcSec::Service_AA . . . . .	112
Arc::Service_JavaWrapper . . . . .	113
Arc::Service_PythonWrapper . . . . .	114
ArcSec::Service_SLCS . . . . .	115
SPService::Service_SP . . . . .	116
SimpleMap . . . . .	117
ARex::StagingConfig . . . . .	118
UnixMap . . . . .	119
ARex::UrlMapConfig . . . . .	120
gridftpd::UrlMapConfig . . . . .	121
Arc::UsageReporter . . . . .	122
userspec_t . . . . .	123
voms . . . . .	124
voms_attrs . . . . .	125
ARex::ZeroUInt . . . . .	126





## Chapter 2

# Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">Arc::ApelDestination</a>	9
<a href="#">ARex::ARexGMConfig</a>	11
<a href="#">ARex::ARexJob</a>	12
<a href="#">ARex::ARexService</a>	16
<a href="#">AuthEvaluator</a>	17
<a href="#">AuthUser</a>	18
<a href="#">AuthVO</a>	19
<a href="#">ARex::CacheConfig</a>	20
<a href="#">ARex::CacheConfigException</a>	21
<a href="#">Cache::CacheService</a>	22
<a href="#">Cache::CacheServiceGenerator</a> (DTR Generator for the cache service )	24
<a href="#">Arc::CARDestination</a>	26
<a href="#">ARex::CommFIFO</a>	27
<a href="#">gridftp::ConfigSections</a>	28
<a href="#">ARex::ConfigSections</a>	29
<a href="#">ARex::ContinuationPlugins</a>	30
<a href="#">ARex::CoreConfig</a> (Parses configuration and fills <a href="#">GMConfig</a> with information )	31
<a href="#">ARex::CountedResource</a>	32
<a href="#">gridftp::Daemon</a>	33
<a href="#">DataStaging::DataDeliveryService</a> (Service for the Delivery layer of data staging )	34
<a href="#">ARex::DelegationStore</a>	35
<a href="#">ARex::DelegationStores</a> (Set of service storing delegated credentials )	36
<a href="#">Arc::Destination</a>	37
<a href="#">Arc::Destinations</a>	38
<a href="#">DirectAccess::diraccess_t</a>	39
<a href="#">DirectAccess</a>	40
<a href="#">DirectFilePlugin</a>	41
<a href="#">DirEntry</a>	42
<a href="#">ARex::DTRGenerator</a>	43
<a href="#">ARex::DTRInfo</a>	46
<a href="#">Entry</a>	47
<a href="#">ARex::Exec</a>	48
<a href="#">ARex::FileChunks</a> (Representation of delivered file chunks )	49

ARex::FileChunksList (Container for <a href="#">FileChunks</a> instances )	50
ARex::FileChunksRef	51
ARex::FileData	52
FileNode	53
FilePlugin	54
ARex::FileRecord	55
FileRoot	56
GACLPlugin	57
gm_dirs_	58
ARex::GMConfig (Configuration information related to the grid manager part of A-REX )	59
gridftpd::GMEEnvironment	62
ARex::GMJob (Represents a job in memory as it passes through the <a href="#">JobsList</a> state machine )	63
GridFTP_Commands	64
GridFTP_Commands_timeout	65
ARex::GridManager	66
Identity	67
IdentityGACL	68
IdentityItemDN	69
IdentityItemVOMS	70
Index	71
ObjectAccess::Item	72
Identity::Item	73
ARex::FileRecord::Iterator	74
ARex::job_state_rec_t	75
ARex::JobDescriptionHandler	76
ARex::JobIDGenerator	78
ARex::JobIDGeneratorARC	79
ARex::JobIDGeneratorES	80
ARex::JobLocalDescription	81
ARex::JobLog	82
Arc::JobLogFile	83
JobPlugin	85
ARex::JobReqResult (Return value of parsing operation )	86
ARex::JobsList	87
gridftpd::LdapQuery	88
gridftpd::LdapQueryError	90
ARex::RunPlugin::lib_plugin_t	91
gridftpd::RunPlugin::lib_plugin_t	92
ARex::LRMSResult	93
Arc::LutsDestination	94
ObjectAccess	95
ObjectAccessGACL	96
ARex::OptimizedInformationContainer	97
gridftpd::ParallelLdapQueries	98
ARex::PayloadBigFile	99
ARex::PayloadFAFile	100
ARex::PayloadFile	101
Permission	102
PermissionGACL	103
Policy	104
ARex::ContinuationPlugins::result_t	105
ARex::RunParallel (Run child process in parallel with stderr redirected to job.jobid.errors )	106
ARex::RunPlugin (Run external process for acquiring local credentials )	107
gridftpd::RunPlugin	108

<a href="#">ARex::RunRedirected</a> (Run child process with stdin, stdout and stderr redirected to specified handles ) . . . . .	109
<a href="#">Server</a> . . . . .	110
<a href="#">FileRoot::ServerParams</a> . . . . .	111
<a href="#">ArcSec::Service_AA</a> . . . . .	112
<a href="#">Arc::Service_JavaWrapper</a> . . . . .	113
<a href="#">Arc::Service_PythonWrapper</a> . . . . .	114
<a href="#">ArcSec::Service_SLCS</a> . . . . .	115
<a href="#">SPService::Service_SP</a> . . . . .	116
<a href="#">SimpleMap</a> . . . . .	117
<a href="#">ARex::StagingConfig</a> (Represents configuration of DTR data staging ) . . . . .	118
<a href="#">UnixMap</a> . . . . .	119
<a href="#">ARex::UrlMapConfig</a> . . . . .	120
<a href="#">gridftpd::UrlMapConfig</a> . . . . .	121
<a href="#">Arc::UsageReporter</a> . . . . .	122
<a href="#">userspec_t</a> . . . . .	123
<a href="#">voms</a> . . . . .	124
<a href="#">voms_attrs</a> . . . . .	125
<a href="#">ARex::ZeroUInt</a> . . . . .	126

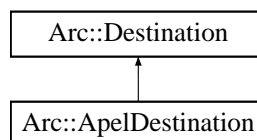


## Chapter 3

# 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 ARex::CacheConfig Class Reference

```
#include <CacheConfig.h>
```

### Public Member Functions

- [CacheConfig](#) (const [GMConfig](#) &config)
- [CacheConfig](#) (const Arc::XMLNode &cfg)
- [CacheConfig](#) ()
- void [substitute](#) (const [GMConfig](#) &config, const Arc::User &user)

### 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 ARex::CacheConfig::CacheConfig (const GMConfig & config)

Create a new [CacheConfig](#) instance. Read the config file and fill in private member variables with cache parameters.

#### 3.8.2.2 ARex::CacheConfig::CacheConfig (const Arc::XMLNode & cfg)

Create a new [CacheConfig](#) instance. Read the XML config tree and fill in private member variables with cache parameters.

#### 3.8.2.3 ARex::CacheConfig::CacheConfig () [inline]

Empty [CacheConfig](#)

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

- [CacheConfig.h](#)



## 3.9 ARex::CacheConfigException Class Reference

```
#include <CacheConfig.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:

- CacheConfig.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 Arc::User &mapped\_user)
- Arc::MCC\_Status [CacheLink](#) (Arc::XMLNode in, Arc::XMLNode out, 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 Arc::User & *mapped\_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:**

*mapped\_user* The local user to which the client DN was mapped

### 3.10.3.2 Arc::MCC\_Status Cache::CacheService::CacheLink (Arc::XMLNode in, Arc::XMLNode out, 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:**

*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 [ARex::GMConfig](#) &config, bool with\_alex)
- [~CacheServiceGenerator](#) ()
- void [receiveDTR](#) (DataStaging::DTR\_ptr dtr)
- bool [addNewRequest](#) (const Arc::User &user, const std::string &source, const std::string &destination, const Arc::UserConfig &usercfg, const std::string &jobid, 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 ARex::GMConfig &config, bool with\_alex)

Start Generator and get Scheduler instance. If with\_alex 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:

*config* A-REX configuration

*with\_alex* 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 Arc::User &user, const std::string &source, const std::string &destination, const Arc::UserConfig &usercfg, const std::string &jobid, int priority)

Add a new request.

#### Parameters:

*user* User for this transfer

*source* Source file

*destination* Destination file

*usercfg* UserConfig with proxy information

*jobid* Job identifier

*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:

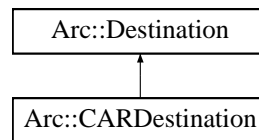
True if all requests for the job have finished, false otherwise

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 ARex::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:

- `conf_sections.h`



## 3.15 ARex::ConfigSections Class Reference

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

- ConfigSections.h

## 3.16 ARex::ContinuationPlugins Class Reference

### Data Structures

- class `command_t`
- class `result_t`

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

- ContinuationPlugins.h

## 3.17 ARex::CoreConfig Class Reference

Parses configuration and fills [GMConfig](#) with information.

```
#include <CoreConfig.h>
```

### Static Public Member Functions

- static bool [ParseConf](#) ([GMConfig](#) &config)

#### 3.17.1 Detailed Description

Parses configuration and fills [GMConfig](#) with information.

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

- CoreConfig.h

### 3.18 ARex::CountedResource Class Reference

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

- arex.h

## 3.19 gridftpd::Daemon Class Reference

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

- daemon.h

## 3.20 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)

### 3.20.1 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

## 3.21 ARex::DelegationStore Class Reference

### Data Structures

- class **Consumer**

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

- DelegationStore.h

## 3.22 ARex::DelegationStores Class Reference

Set of service storing delegated credentials.

```
#include <DelegationStores.h>
```

### Public Member Functions

- [DelegationStore](#) & [operator\[\]](#) (const std::string &path)
- bool [MatchNamespace](#) (const Arc::SOAPEnvelope &in)
- bool [Process](#) (const std::string &path, const Arc::SOAPEnvelope &in, Arc::SOAPEnvelope &out, const std::string &client, std::string &credentials)
- bool [DelegatedToken](#) (const std::string &path, Arc::XMLNode token, const std::string &client, std::string &credentials)

### 3.22.1 Detailed Description

Set of service storing delegated credentials.

### 3.22.2 Member Function Documentation

#### 3.22.2.1 bool ARex::DelegationStores::DelegatedToken (const std::string & *path*, Arc::XMLNode *token*, const std::string & *client*, std::string & *credentials*)

Stores delegated credentials token defined by 'token' into storage 'path'. Extracted token is also returned in 'credentials'. If operation is successful returns true.

#### 3.22.2.2 bool ARex::DelegationStores::Process (const std::string & *path*, const Arc::SOAPEnvelope & *in*, Arc::SOAPEnvelope & *out*, const std::string & *client*, std::string & *credentials*)

Processes SOAP request 'in' using delegation storage associated with 'path'. Response is filled into 'out'. The 'client' is identifier of requestor used by service internally to recognize owner of stored credentials. If operation produces credentials token it is returned in 'credentials'. If operation is successful returns true.

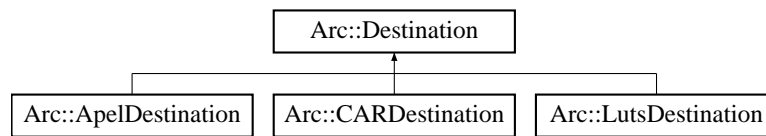
The documentation for this class was generated from the following file:

- DelegationStores.h



## 3.23 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)

#### 3.23.1 Detailed Description

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

#### 3.23.2 Member Function Documentation

##### 3.23.2.1 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.

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

Finishes pending submission of records.

Reimplemented in [Arc::ApelDestination](#), [Arc::CARDestination](#), and [Arc::LutsDestination](#).

##### 3.23.2.3 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

## 3.24 Arc::Destinations Class Reference

```
#include <Destinations.h>
```

### Public Member Functions

- void [report](#) ([Arc::JobLogFile](#) &joblog)

### 3.24.1 Detailed Description

Class to handle a set of reporting destinations.

### 3.24.2 Member Function Documentation

#### 3.24.2.1 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

## 3.25 DirectAccess::diraccess\_t Struct Reference

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

- fileplugin.h

## 3.26 DirectAccess Class Reference

### Data Structures

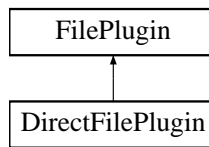
- struct [diraccess\\_t](#)

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

- fileplugin.h

## 3.27 DirectFilePlugin Class Reference

Inheritance diagram for DirectFilePlugin::



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

- fileplugin.h

## 3.28 DirEntry Class Reference

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

- fileroot.h

## 3.29 ARex::DTRGenerator Class Reference

```
#include <DTRGenerator.h>
```

### Public Member Functions

- **DTRGenerator** (const **GMConfig** &config, void(\*kicker\_func)(void \*)=NULL, void \*kicker\_arg=NULL)
- **~DTRGenerator** ()
- virtual void **receiveDTR** (DataStaging::DTR\_ptr dtr)
- void **receiveJob** (const **GMJob** &job)
- void **cancelJob** (const **GMJob** &job)
- bool **queryJobFinished** (**GMJob** &job)
- bool **hasJob** (const **GMJob** &job)
- void **removeJob** (const **GMJob** &job)
- int **checkUploadedFiles** (**GMJob** &job)

### 3.29.1 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.

### 3.29.2 Constructor & Destructor Documentation

#### 3.29.2.1 ARex::DTRGenerator::DTRGenerator (const GMConfig & config, void(\*) (void \*) kicker\_func = NULL, void \* kicker\_arg = NULL)

Start up Generator.

##### Parameters:

- user* Grid manager configuration.
- kicker\_func* Function to call on completion of all DTRs for a job
- kicker\_arg* Argument to kicker function

#### 3.29.2.2 ARex::DTRGenerator::~~DTRGenerator ()

Stop Generator

### 3.29.3 Member Function Documentation

#### 3.29.3.1 void ARex::DTRGenerator::cancelJob (const GMJob & 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 asynchronously deals with cancelling the DTRs.

##### Parameters:

- job* The job which is being cancelled

### 3.29.3.2 int ARex::DTRGenerator::checkUploadedFiles (GMJob & *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

### 3.29.3.3 bool ARex::DTRGenerator::hasJob (const GMJob & *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.

### 3.29.3.4 bool ARex::DTRGenerator::queryJobFinished (GMJob & *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.

### 3.29.3.5 virtual void ARex::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



### 3.29.3.6 void ARex::DTRGenerator::receiveJob (const GMJob & *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.

### 3.29.3.7 void ARex::DTRGenerator::removeJob (const GMJob & *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:

- DTRGenerator.h

## 3.30 ARex::DTRInfo Class Reference

```
#include <DTRGenerator.h>
```

### 3.30.1 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.

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

- DTRGenerator.h

## 3.31 Entry Class Reference

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

- Entry.h

## 3.32 ARex::Exec Class Reference

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

- ControlFileContent.h

## 3.33 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)

### 3.33.1 Detailed Description

Representation of delivered file chunks.

### 3.33.2 Member Function Documentation

#### 3.33.2.1 void ARex::FileChunks::Release (void)

Release reference obtained through [FileChunksList::Get\(\)](#) method. This operation may lead to destruction of FileChunk instance hence previously obtained reference must not be used.

#### 3.33.2.2 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](#)

## 3.34 ARex::FileChunksList Class Reference

Container for [FileChunks](#) instances.

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

### Public Member Functions

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

#### 3.34.1 Detailed Description

Container for [FileChunks](#) instances.

#### 3.34.2 Member Function Documentation

##### 3.34.2.1 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

## 3.35 ARex::FileChunksRef Class Reference

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

- FileChunks.h

### 3.36 ARex::FileData Class Reference

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

- ControlFileContent.h



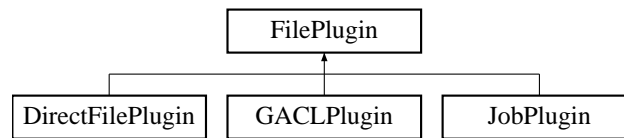
## 3.37 FileNode Class Reference

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

- fileroot.h

### 3.38 FilePlugin Class Reference

Inheritance diagram for FilePlugin::



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

- fileroot.h

## 3.39 ARex::FileRecord Class Reference

### Data Structures

- class [Iterator](#)

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

- FileRecord.h

## 3.40 FileRoot Class Reference

### Data Structures

- class [ServerParams](#)

### Friends

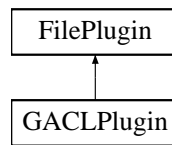
- class [GridFTP\\_Commands](#)

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

- fileroot.h

## 3.41 GACLPlugin Class Reference

Inheritance diagram for GACLPlugin::



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

- gacplugin.h

## 3.42 gm\_dirs\_ Struct Reference

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

- `jobplugin.h`

## 3.43 ARex::GMConfig Class Reference

Configuration information related to the grid manager part of A-REX.

```
#include <GMConfig.h>
```

### Data Structures

- class **ExternalHelper**  
*Class to run external processes (helper).*

### Public Types

- enum [fixdir\\_t](#)

### Public Member Functions

- [GMConfig](#) (const std::string &conffile="")
- [GMConfig](#) (const Arc::XMLNode &node)
- bool [Load](#) ()
- void [Print](#) () const
- const std::string & [ConfigFile](#) () const
- void [SetConfigFile](#) (const std::string &file)
- bool [ConfigIsTemp](#) () const
- void [SetConfigIsTemp](#) (bool temp)
- void [SetXMLNode](#) (const Arc::XMLNode &node)
- bool [CreateControlDirectory](#) () const
- bool [CreateSessionDirectory](#) (const std::string &dir, const Arc::User &user) const
- bool [RunHelpers](#) ()
- bool [Substitute](#) (std::string &param, const Arc::User &user=Arc::User()) const
- void [PrepareToDestroy](#) ()
- void [SetControlDir](#) (const std::string &dir)
- void [SetSessionRoot](#) (const std::string &dir)
- void [SetSessionRoot](#) (const std::vector< std::string > &dirs)
- void [SetShareID](#) (const Arc::User &share\_user)
- void [SetDefaultQueue](#) (const std::string &queue)
- const std::string & [CertDir](#) () const
- const std::string & [VomsDir](#) () const
- const std::string & [RTEDir](#) () const
- std::string [DelegationDir](#) () const
- const std::string & [SupportMailAddress](#) () const
- void [SetJobLog](#) (JobLog \*log)
- void [SetContPlugins](#) (ContinuationPlugins \*plugins)
- void [SetCredPlugin](#) (RunPlugin \*plugin)
- void [SetDelegations](#) (ARex::DelegationStores \*stores)
- JobLog \* [GetJobLog](#) () const
- ContinuationPlugins \* [ContPlugins](#) () const
- RunPlugin \* [CredPlugin](#) () const

- [ARex::DelegationStores](#) \* [Delegations](#) () const
- std::string [ControlDir](#) () const
- std::string [SessionRoot](#) (const std::string &job\_id) const
- std::vector< std::string > [SessionRoots](#) () const
- std::vector< std::string > [SessionRootsNonDraining](#) () const
- std::string [ScratchDir](#) () const
- bool [StrictSession](#) () const
- [CacheConfig](#) [CacheParams](#) () const
- const std::string & [HeadNode](#) () const
- bool [ARCInterfaceEnabled](#) () const
- bool [EMIESInterfaceEnabled](#) () const
- const std::string & [GridFTPEndpoint](#) () const
- const std::string & [AREXEndpoint](#) () const
- const std::string & [DefaultLRMS](#) () const
- const std::string & [DefaultQueue](#) () const
- std::list< std::string > [Queues](#) () const
- const std::string & [UnixName](#) () const
- const std::string & [AllowSubmit](#) () const
- time\_t [KeepFinished](#) () const
- time\_t [KeepDeleted](#) () const
- int [Reruns](#) () const
- [fixdir\\_t](#) [FixDirectories](#) () const
- unsigned int [WakeupPeriod](#) () const
- int [MaxJobs](#) () const
- int [MaxRunning](#) () const
- int [MaxPerDN](#) () const
- int [MaxTotal](#) () const
- int [MaxJobsStaging](#) () const
- int [MaxStagingEmergency](#) () const
- int [MaxDownloads](#) () const
- bool [UseDTR](#) () const
- bool [MatchShareUid](#) (uid\_t suid) const
- bool [MatchShareGid](#) (gid\_t sgid) const

### 3.43.1 Detailed Description

Configuration information related to the grid manager part of A-REX. This class contains all configuration variables related to grid-manager. It also acts as a container for objects which are used in different parts of A-REX. Therefore since this class contains pointers to complex objects, it cannot be copied and hence the copy constructor and assignment operator are private. Those pointers should be managed outside this class. [GMConfig](#) should be instantiated once when the grid-manager is initialised and only destroyed with the GM has finished. Ideally this would be a singleton but that would prevent running multiple A-REXes in the same container.

Substitutions are not done while parsing the configuration, as substitution variables can change depending on the job. Therefore paths are stored in their raw format, unsubstituted. The exception is the control directory which cannot change and is substituted during parsing, and helper options. Substitution of other variables should be done as necessary using [Substitute\(\)](#).



## 3.43.2 Constructor & Destructor Documentation

### 3.43.2.1 ARex::GMConfig::GMConfig (const std::string & *conf*file = "")

Use given (or guessed if not given) configuration file. Guessing uses \$ARC\_CONFIG, \$ARC\_LOCATION/etc/arc.conf or the default location /etc/arc.conf. [Load\(\)](#) should then be used to parse the configuration and fill member variables.

#### Parameters:

*conf*file Path to configuration file, will be guessed if empty

## 3.43.3 Member Function Documentation

### 3.43.3.1 bool ARex::GMConfig::CreateControlDirectory () const

Create control structure with permissions depending on fixdir\_t value. Typically called at A-REX service creation.

### 3.43.3.2 bool ARex::GMConfig::CreateSessionDirectory (const std::string & *dir*, const Arc::User & *user*) const

Create session directory with correct permissions. Typically called when a new job is created and after all substitutions have been done. Creates session root if it does not already exist.

### 3.43.3.3 bool ARex::GMConfig::Load ()

Load configuration from file or XML node into members of this object. Returns false if errors are found during parsing.

### 3.43.3.4 std::string ARex::GMConfig::SessionRoot (const std::string & *job\_id*) const

Session root directory corresponding to given job ID. If the session dir corresponding to job\_id is not found an empty string is returned.

### 3.43.3.5 bool ARex::GMConfig::Substitute (std::string & *param*, const Arc::User & *user* = Arc::User ()) const

Substitute characters in param specified by % with real values. An optional User can be specified for the user-related substitutions.

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

- GMConfig.h

## 3.44 gridftp::GMEEnvironment Class Reference

### Public Member Functions

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

### 3.44.1 Member Function Documentation

#### 3.44.1.1 `std::string gridftp::GMEEnvironment::nordugrid_config_loc` (void) const

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

#### 3.44.1.2 `std::string gridftp::GMEEnvironment::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:

- `environment.h`

## 3.45 ARex::GMJob Class Reference

Represents a job in memory as it passes through the [JobsList](#) state machine.

```
#include <GMJob.h>
```

### 3.45.1 Detailed Description

Represents a job in memory as it passes through the [JobsList](#) state machine.

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

- GMJob.h

## 3.46 GridFTP\_Commands Class Reference

### Data Structures

- class `close_semaphore_t`
- struct `data_buffer_t`

### Friends

- class [GridFTP\\_Commands\\_timeout](#)

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

- `commands.h`

## 3.47 GridFTP\_Commands\_timeout Class Reference

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

- `commands.h`

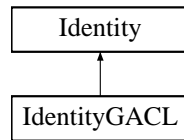
## 3.48 ARex::GridManager Class Reference

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

- GridManager.h

## 3.49 Identity Class Reference

Inheritance diagram for Identity::



### Data Structures

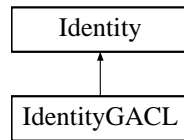
- class [Item](#)

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

- identity.h

## 3.50 IdentityGACL Class Reference

Inheritance diagram for IdentityGACL::



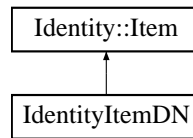
The documentation for this class was generated from the following file:

- identity\_gacl.h



## 3.51 IdentityItemDN Class Reference

Inheritance diagram for IdentityItemDN::

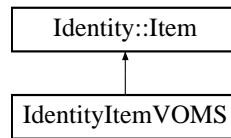


The documentation for this class was generated from the following file:

- identity\_dn.h

## 3.52 IdentityItemVOMS Class Reference

Inheritance diagram for IdentityItemVOMS::



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

- identity\_voms.h

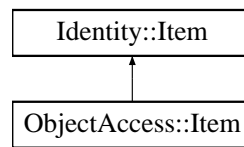
## 3.53 Index Class Reference

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

- Index.h

### 3.54 ObjectAccess::Item Class Reference

Inheritance diagram for ObjectAccess::Item::

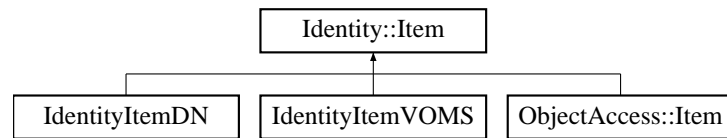


The documentation for this class was generated from the following file:

- object\_access.h

## 3.55 Identity::Item Class Reference

Inheritance diagram for Identity::Item::



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

- identity.h

## 3.56 ARex::FileRecord::Iterator Class Reference

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

- FileRecord.h

## 3.57 ARex::job\_state\_rec\_t Struct Reference

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

- GMJob.h

## 3.58 ARex::JobDescriptionHandler Class Reference

```
#include <JobDescriptionHandler.h>
```

### Data Structures

- class **value\_for\_shell**

*Class for handling escapes and quotes when writing to .grami.*

### Public Member Functions

- [JobDescriptionHandler](#) (const [GMConfig](#) &config)
- [JobReqResult](#) [parse\\_job\\_req](#) ([JobLocalDescription](#) &job\_desc, [Arc::JobDescription](#) &arc\_job\_desc, const std::string &fname, bool check\_acl=false) const
- [JobReqResult](#) [parse\\_job\\_req](#) (const [JobId](#) &job\_id, [JobLocalDescription](#) &job\_desc, bool check\_acl=false) const
- [JobReqResult](#) [parse\\_job\\_req](#) (const [JobId](#) &job\_id, [JobLocalDescription](#) &job\_desc, [Arc::JobDescription](#) &arc\_job\_desc, bool check\_acl=false) const
- bool [process\\_job\\_req](#) (const [GMJob](#) &job, [JobLocalDescription](#) &job\_desc) const
- bool [write\\_grami](#) (const [GMJob](#) &job, const char \*opt\_add=NULL) const
- bool [write\\_grami](#) (const [Arc::JobDescription](#) &arc\_job\_desc, const [GMJob](#) &job, const char \*opt\_add) const
- std::string [get\\_local\\_id](#) (const [JobId](#) &job\_id) const
- bool [set\\_execs](#) (const [GMJob](#) &job) const

### 3.58.1 Detailed Description

Deals with parsing and converting job descriptions between [Arc::JobDescription](#) and [JobLocalDescription](#). Also deals with reading and writing .grami file.

### 3.58.2 Member Function Documentation

#### 3.58.2.1 [JobReqResult](#) [ARex::JobDescriptionHandler::parse\\_job\\_req](#) (const [JobId](#) &job\_id, [JobLocalDescription](#) &job\_desc, [Arc::JobDescription](#) &arc\_job\_desc, bool check\_acl = false) const

Parse the job description for job\_id into job\_desc and arc\_job\_desc. Optionally check acl file and put result into returned object

#### 3.58.2.2 [JobReqResult](#) [ARex::JobDescriptionHandler::parse\\_job\\_req](#) (const [JobId](#) &job\_id, [JobLocalDescription](#) &job\_desc, bool check\_acl = false) const

Parse the job description for job\_id into job\_desc. Optionally check acl file and put result into returned object



### 3.58.2.3 JobReqResult ARex::JobDescriptionHandler::parse\_job\_req (JobLocalDescription & *job\_desc*, Arc::JobDescription & *arc\_job\_desc*, const std::string & *fname*, bool *check\_acl* = **false**) const

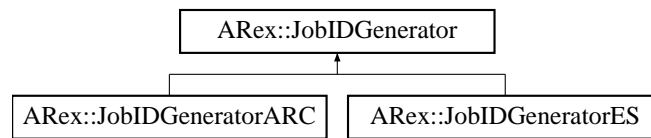
Parse the job description at the given file into *job\_desc* and *arc\_job\_desc*. Optionally check acl file and put result into returned object

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

- JobDescriptionHandler.h

### 3.59 ARex::JobIDGenerator Class Reference

Inheritance diagram for ARex::JobIDGenerator::

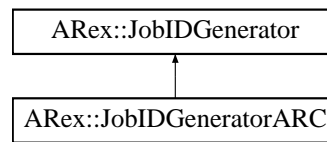


The documentation for this class was generated from the following file:

- tools.h

## 3.60 ARex::JobIDGeneratorARC Class Reference

Inheritance diagram for ARex::JobIDGeneratorARC::

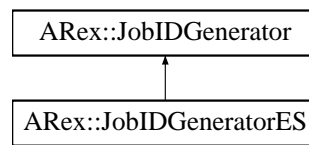


The documentation for this class was generated from the following file:

- tools.h

### 3.61 ARex::JobIDGeneratorES Class Reference

Inheritance diagram for ARex::JobIDGeneratorES::



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

- tools.h

## 3.62 ARex::JobLocalDescription Class Reference

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

- ControlFileContent.h

## 3.63 ARex::JobLog Class Reference

```
#include <JobLog.h>
```

### 3.63.1 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:

- JobLog.h

## 3.64 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](#) ()

### 3.64.1 Detailed Description

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

### 3.64.2 Constructor & Destructor Documentation

#### 3.64.2.1 Arc::JobLogFile::JobLogFile (const std::string &\_filename) [inline]

Constructor. Loads and parses A-REX job log.

References [parse\(\)](#).

### 3.64.3 Member Function Documentation

#### 3.64.3.1 void Arc::JobLogFile::allowRemove (bool a) [inline]

Enables/disables file removal from disk

#### 3.64.3.2 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.

#### 3.64.3.3 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.

#### **3.64.3.4    `bool Arc::JobLogFile::exists ()`**

Checks if file exists on the disk

#### **3.64.3.5    `std::string Arc::JobLogFile::getFilename ()    [inline]`**

Returns original full path to log file

#### **3.64.3.6    `bool Arc::JobLogFile::olderThan (time_t age)`**

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

#### **3.64.3.7    `int Arc::JobLogFile::parse (const std::string & _filename)`**

Reloads and parses A-REX job log.

Referenced by JobLogFile().

#### **3.64.3.8    `void Arc::JobLogFile::remove ()`**

Deletes file from the disk

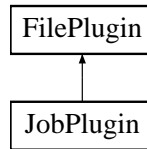
The documentation for this class was generated from the following file:

- jura/JobLogFile.h



## 3.65 JobPlugin Class Reference

Inheritance diagram for JobPlugin::



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

- jobplugin.h

## 3.66 ARex::JobReqResult Class Reference

Return value of parsing operation.

```
#include <JobDescriptionHandler.h>
```

### 3.66.1 Detailed Description

Return value of parsing operation.

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

- JobDescriptionHandler.h

## 3.67 ARex::JobsList Class Reference

```
#include <JobsList.h>
```

### 3.67.1 Detailed Description

List of jobs. This class contains the main job management logic which moves jobs through the state machine. New jobs found through Scan methods are held in memory until reaching FINISHED state.

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

- JobsList.h

## 3.68 gridftp::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](#) ()

### 3.68.1 Detailed Description

[LdapQuery](#) class; querying of LDAP servers.

### 3.68.2 Member Enumeration Documentation

#### 3.68.2.1 enum gridftp::LdapQuery::Scope

Scope for a LDAP queries. Use when querying.

### 3.68.3 Constructor & Destructor Documentation

#### 3.68.3.1 gridftp::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](#).

#### 3.68.3.2 gridftp::LdapQuery::~~LdapQuery ()

Destructor. Will disconnect from the ldapservers if still connected.

### 3.68.4 Member Function Documentation

#### 3.68.4.1 std::string gridftp::LdapQuery::Host ()

Returns the hostname of the ldap-server.

**3.68.4.2** void gridftp::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.

**3.68.4.3** void gridftp::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

## 3.69 gridftp::LdapQueryError Class Reference

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

### Public Member Functions

- [LdapQueryError](#) (std::string message)

#### 3.69.1 Detailed Description

[LdapQuery](#) exception. Gets thrown when an error occurs in a query.

#### 3.69.2 Constructor & Destructor Documentation

##### 3.69.2.1 gridftp::LdapQueryError::LdapQueryError (std::string *message*) [inline]

Standard exception class constructor.

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

- ldapquery.h

## 3.70 ARex::RunPlugin::lib\_plugin\_t Union Reference

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

- RunPlugin.h

### 3.71 `gridftpd::RunPlugin::lib_plugin_t` Union Reference

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

- `run_plugin.h`



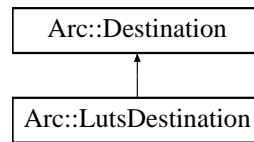
## 3.72 ARex::LRMSResult Class Reference

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

- ControlFileContent.h

## 3.73 Arc::LutsDestination Class Reference

`#include <LutsDestination.h>`Inheritance diagram for Arc::LutsDestination::



### Public Member Functions

- [LutsDestination](#) ([JobLogFile](#) &joblog)
- void [report](#) ([JobLogFile](#) &joblog)
- void [finish](#) ()

#### 3.73.1 Detailed Description

Reporting destination adapter for SGAS LUTS.

#### 3.73.2 Constructor & Destructor Documentation

##### 3.73.2.1 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.

#### 3.73.3 Member Function Documentation

##### 3.73.3.1 void Arc::LutsDestination::finish () [virtual]

Finishes pending submission of records.

Reimplemented from [Arc::Destination](#).

##### 3.73.3.2 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.

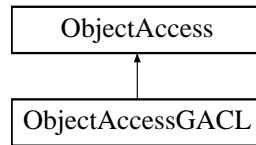
Implements [Arc::Destination](#).

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

- `LutsDestination.h`

## 3.74 ObjectAccess Class Reference

Inheritance diagram for ObjectAccess::



### Data Structures

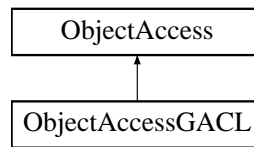
- class [Item](#)

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

- `object_access.h`

## 3.75 ObjectAccessGACL Class Reference

Inheritance diagram for ObjectAccessGACL::



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

- object\_access\_gacl.h

## 3.76 ARex::OptimizedInformationContainer Class Reference

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

- arex.h

## 3.77 gridftpd::ParallelLdapQueries Class Reference

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

### 3.77.1 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

## 3.78 ARex::PayloadBigFile Class Reference

### Public Member Functions

- [PayloadBigFile](#) (const char \*filename, Size\_t start, Size\_t end)
- virtual [~PayloadBigFile](#) (void)

### 3.78.1 Constructor & Destructor Documentation

#### 3.78.1.1 ARex::PayloadBigFile::PayloadBigFile (const char \* *filename*, Size\_t *start*, Size\_t *end*)

Creates object associated with file for reading from it

#### 3.78.1.2 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

## 3.79 ARex::PayloadFAFile Class Reference

### Public Member Functions

- [PayloadFAFile](#) (Arc::FileAccess \*h, Size\_t start, Size\_t end)

### 3.79.1 Constructor & Destructor Documentation

#### 3.79.1.1 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



## 3.80 ARex::PayloadFile Class Reference

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

### Public Member Functions

- [PayloadFile](#) (const char \*filename, Size\_t start, Size\_t end)
- virtual [~PayloadFile](#) (void)

### 3.80.1 Detailed Description

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

### 3.80.2 Constructor & Destructor Documentation

#### 3.80.2.1 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.

#### 3.80.2.2 virtual ARex::PayloadFile::~~PayloadFile (void) [virtual]

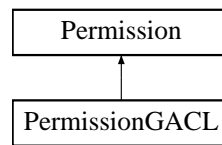
Creates object associated with file for writing into it. Use size=-1 for undefined size.

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

- PayloadFile.h

## 3.81 Permission Class Reference

Inheritance diagram for Permission::

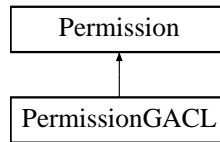


The documentation for this class was generated from the following file:

- permission.h

## 3.82 PermissionGACL Class Reference

Inheritance diagram for PermissionGACL::



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

- permission\_gacl.h

## 3.83 Policy Class Reference

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

- Policy.h

## 3.84 ARex::ContinuationPlugins::result\_t Class Reference

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

- ContinuationPlugins.h

## 3.85 ARex::RunParallel Class Reference

Run child process in parallel with stderr redirected to job.jobid.errors.

```
#include <RunParallel.h>
```

### 3.85.1 Detailed Description

Run child process in parallel with stderr redirected to job.jobid.errors.

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

- RunParallel.h

## 3.86 ARex::RunPlugin Class Reference

Run external process for acquiring local credentials.

```
#include <RunPlugin.h>
```

### Data Structures

- union [lib\\_plugin\\_t](#)

#### 3.86.1 Detailed Description

Run external process for acquiring local credentials.

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

- RunPlugin.h

## 3.87 gridftpd::RunPlugin Class Reference

### Data Structures

- union [lib\\_plugin\\_t](#)

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

- [run\\_plugin.h](#)



## 3.88 ARex::RunRedirected Class Reference

Run child process with stdin, stdout and stderr redirected to specified handles.

```
#include <RunRedirected.h>
```

### 3.88.1 Detailed Description

Run child process with stdin, stdout and stderr redirected to specified handles.

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

- RunRedirected.h

## 3.89 Server Class Reference

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

- `Server.h`

## 3.90 FileRoot::ServerParams Class Reference

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

- fileroot.h

## 3.91 ArcSec::Service\_AA Class Reference

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

### 3.91.1 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

## 3.92 Arc::Service\_JavaWrapper Class Reference

### Public Member Functions

- virtual Arc::MCC\_Status [process](#) (Arc::Message &, Arc::Message &)

### 3.92.1 Member Function Documentation

#### 3.92.1.1 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

## 3.93 Arc::Service\_PythonWrapper Class Reference

### Public Member Functions

- virtual Arc::MCC\_Status [process](#) (Arc::Message &, Arc::Message &)

### 3.93.1 Member Function Documentation

#### 3.93.1.1 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

## 3.94 ArcSec::Service\_SLCS Class Reference

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

### 3.94.1 Detailed Description

A Service which signs the short-lived certificate; it accepts the certificate signing request (CSR) 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 conversion from username/password -----> x509 certificate.

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

- slcs.h

## 3.95 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 &)

### 3.95.1 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 directly when it gives a request, instead of the WRYF(where are you from) or Discovery Service in other implementation

### 3.95.2 Constructor & Destructor Documentation

#### 3.95.2.1 SPSERVICE::Service\_SP::Service\_SP (Arc::Config \* *cfg*)

Constructor

### 3.95.3 Member Function Documentation

#### 3.95.3.1 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



## 3.96 SimpleMap Class Reference

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

- `simplemap.h`

## 3.97 ARex::StagingConfig Class Reference

Represents configuration of DTR data staging.

```
#include <StagingConfig.h>
```

### Public Member Functions

- [StagingConfig](#) (const [GMConfig](#) &config)

### 3.97.1 Detailed Description

Represents configuration of DTR data staging.

### 3.97.2 Constructor & Destructor Documentation

#### 3.97.2.1 ARex::StagingConfig::StagingConfig (const GMConfig & config)

Load config from configuration file. Information from [GMConfig](#) 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:

- StagingConfig.h

## 3.98 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

### 3.99 ARex::UrlMapConfig Class Reference

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

- UrlMapConfig.h

## 3.100 gridftpd::UrlMapConfig Class Reference

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

- conf\_map.h

## 3.101 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](#) ()

### 3.101.1 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.

### 3.101.2 Constructor & Destructor Documentation

**3.101.2.1** `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).

### 3.101.3 Member Function Documentation

**3.101.3.1** `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

## 3.102 userspec\_t Class Reference

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

- userspec.h

## 3.103 voms Struct Reference

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

### Data Fields

- std::string [server](#)
- std::string [voname](#)
- std::vector< [voms\\_attrs](#) > [attrs](#)

### 3.103.1 Detailed Description

VOMS data

### 3.103.2 Field Documentation

#### 3.103.2.1 std::vector<voms\_attrs> voms::attrs

User's characteristics

#### 3.103.2.2 std::string voms::server

The VOMS server DN, as from its certificate

#### 3.103.2.3 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](#)



## 3.104 voms\_attrs Struct Reference

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

### Data Fields

- std::string [group](#)
- std::string [role](#)
- std::string [cap](#)

### 3.104.1 Detailed Description

VOMS attributes

### 3.104.2 Field Documentation

#### 3.104.2.1 std::string voms\_attrs::cap

user's capability

#### 3.104.2.2 std::string voms\_attrs::group

user's group

#### 3.104.2.3 std::string voms\_attrs::role

user's role

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

- [auth.h](#)

## 3.105 ARex::ZeroUInt Class Reference

```
#include <JobsList.h>
```

### 3.105.1 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. TODO: move to common lib?

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

- JobsList.h

# Index

- ~CacheService
  - Cache::CacheService, [22](#)
- ~DTRGenerator
  - ARex::DTRGenerator, [43](#)
- ~LdapQuery
  - gridftp::LdapQuery, [88](#)
- ~PayloadBigFile
  - ARex::PayloadBigFile, [99](#)
- ~PayloadFile
  - ARex::PayloadFile, [101](#)
- addNewRequest
  - Cache::CacheServiceGenerator, [24](#)
- allowRemove
  - Arc::JobLogFile, [83](#)
- ApelDestination
  - Arc::ApelDestination, [9](#)
- Arc::ApelDestination, [9](#)
  - ApelDestination, [9](#)
  - finish, [9](#)
  - report, [10](#)
- Arc::CARDestination, [26](#)
  - CARDestination, [26](#)
  - finish, [26](#)
  - report, [26](#)
- Arc::Destination, [37](#)
  - createDestination, [37](#)
  - finish, [37](#)
  - report, [37](#)
- Arc::Destinations, [38](#)
  - report, [38](#)
- Arc::JobLogFile, [83](#)
  - allowRemove, [83](#)
  - createCARUsageRecord, [83](#)
  - createUsageRecord, [83](#)
  - exists, [84](#)
  - getFilename, [84](#)
  - JobLogFile, [83](#)
  - olderThan, [84](#)
  - parse, [84](#)
  - remove, [84](#)
- Arc::LutsDestination, [94](#)
  - finish, [94](#)
  - LutsDestination, [94](#)
  - report, [94](#)
- Arc::Service\_JavaWrapper, [113](#)
  - process, [113](#)
- Arc::Service\_PythonWrapper, [114](#)
  - process, [114](#)
- Arc::UsageReporter, [122](#)
  - report, [122](#)
  - UsageReporter, [122](#)
- ArcSec::Service\_AA, [112](#)
- ArcSec::Service\_SLCS, [115](#)
- ARex::ARexGMConfig, [11](#)
- ARex::ARexJob, [12](#)
  - ARexJob, [12](#)
  - Cancel, [13](#)
  - ChooseSessionDir, [13](#)
  - Clean, [13](#)
  - Created, [13](#)
  - CreateFile, [13](#)
  - Failed, [13](#)
  - FailedState, [13](#)
  - Failure, [13](#)
  - GetDescription, [13](#)
  - ID, [13](#)
  - Jobs, [14](#)
  - LogDir, [14](#)
  - LogFiles, [14](#)
  - Modified, [14](#)
  - OpenDir, [14](#)
  - OpenFile, [14](#)
  - OpenLogFile, [14](#)
  - Resume, [14](#)
  - SessionDir, [14](#)
  - State, [14](#)
  - TotalJobs, [15](#)
  - UpdateCredentials, [15](#)
- ARex::ARexService, [16](#)
- ARex::CacheConfig, [20](#)
  - CacheConfig, [20](#)
- ARex::CacheConfigException, [21](#)
- ARex::CommFIFO, [27](#)
- ARex::ConfigSections, [29](#)
- ARex::ContinuationPlugins, [30](#)
- ARex::ContinuationPlugins::result\_t, [105](#)
- ARex::CoreConfig, [31](#)
- ARex::CountedResource, [32](#)
- ARex::DelegationStore, [35](#)

- ARex::DelegationStores, 36
  - DelegatedToken, 36
  - Process, 36
- ARex::DTRGenerator, 43
  - ~DTRGenerator, 43
  - cancelJob, 43
  - checkUploadedFiles, 43
  - DTRGenerator, 43
  - hasJob, 44
  - queryJobFinished, 44
  - receiveDTR, 44
  - receiveJob, 44
  - removeJob, 45
- ARex::DTRInfo, 46
- ARex::Exec, 48
- ARex::FileChunks, 49
  - Release, 49
  - Remove, 49
- ARex::FileChunksList, 50
  - Get, 50
- ARex::FileChunksRef, 51
- ARex::FileData, 52
- ARex::FileRecord, 55
- ARex::FileRecord::Iterator, 74
- ARex::GMConfig, 59
  - CreateControlDirectory, 61
  - CreateSessionDirectory, 61
  - GMConfig, 61
  - Load, 61
  - SessionRoot, 61
  - Substitute, 61
- ARex::GMJob, 63
- ARex::GridManager, 66
- ARex::job\_state\_rec\_t, 75
- ARex::JobDescriptionHandler, 76
  - parse\_job\_req, 76
- ARex::JobIDGenerator, 78
- ARex::JobIDGeneratorARC, 79
- ARex::JobIDGeneratorES, 80
- ARex::JobLocalDescription, 81
- ARex::JobLog, 82
- ARex::JobReqResult, 86
- ARex::JobsList, 87
- ARex::LRMSResult, 93
- ARex::OptimizedInformationContainer, 97
- ARex::PayloadBigFile, 99
  - ~PayloadBigFile, 99
  - PayloadBigFile, 99
- ARex::PayloadFAFile, 100
  - PayloadFAFile, 100
- ARex::PayloadFile, 101
  - ~PayloadFile, 101
  - PayloadFile, 101
- ARex::RunParallel, 106
- ARex::RunPlugin, 107
- ARex::RunPlugin::lib\_plugin\_t, 91
- ARex::RunRedirected, 109
- ARex::StagingConfig, 118
  - StagingConfig, 118
- ARex::UrlMapConfig, 120
- ARex::ZeroUInt, 126
- ARexJob
  - ARex::ARexJob, 12
- attrs
  - voms, 124
- AuthEvaluator, 17
- AuthUser, 18
- AuthVO, 19
- Cache::CacheService, 22
  - ~CacheService, 22
  - CacheCheck, 22
  - CacheLink, 23
  - CacheLinkQuery, 23
  - CacheService, 22
  - operator bool, 23
  - process, 23
  - RegistrationCollector, 23
- Cache::CacheServiceGenerator, 24
  - addNewRequest, 24
  - CacheServiceGenerator, 24
  - queryRequestsFinished, 24
- CacheCheck
  - Cache::CacheService, 22
- CacheConfig
  - ARex::CacheConfig, 20
- CacheLink
  - Cache::CacheService, 23
- CacheLinkQuery
  - Cache::CacheService, 23
- CacheService
  - Cache::CacheService, 22
- CacheServiceGenerator
  - Cache::CacheServiceGenerator, 24
- Cancel
  - ARex::ARexJob, 13
- cancelJob
  - ARex::DTRGenerator, 43
- cap
  - voms\_attrs, 125
- CARDestination
  - Arc::CARDestination, 26
- checkUploadedFiles
  - ARex::DTRGenerator, 43
- ChooseSessionDir
  - ARex::ARexJob, 13
- Clean
  - ARex::ARexJob, 13

- createCARUsageRecord
  - Arc::JobLogFile, 83
- CreateControlDirectory
  - ARex::GMConfig, 61
- Created
  - ARex::ARexJob, 13
- createDestination
  - Arc::Destination, 37
- CreateFile
  - ARex::ARexJob, 13
- CreateSessionDirectory
  - ARex::GMConfig, 61
- createUsageRecord
  - Arc::JobLogFile, 83
- DataStaging::DataDeliveryService, 34
- DelegatedToken
  - ARex::DelegationStores, 36
- DirectAccess, 40
- DirectAccess::diraccess\_t, 39
- DirectFilePlugin, 41
- DirEntry, 42
- DTRGenerator
  - ARex::DTRGenerator, 43
- Entry, 47
- exists
  - Arc::JobLogFile, 84
- Failed
  - ARex::ARexJob, 13
- FailedState
  - ARex::ARexJob, 13
- Failure
  - ARex::ARexJob, 13
- FileNode, 53
- FilePlugin, 54
- FileRoot, 56
- FileRoot::ServerParams, 111
- finish
  - Arc::ApelDestination, 9
  - Arc::CARDestination, 26
  - Arc::Destination, 37
  - Arc::LutsDestination, 94
- GACLPlugin, 57
- Get
  - ARex::FileChunksList, 50
- GetDescription
  - ARex::ARexJob, 13
- getFilename
  - Arc::JobLogFile, 84
- gm\_dirs\_, 58
- GMConfig
  - ARex::GMConfig, 61
- GridFTP\_Commands, 64
- GridFTP\_Commands\_timeout, 65
- gridftpd::ConfigSections, 28
- gridftpd::Daemon, 33
- gridftpd::GMEEnvironment, 62
  - nordugrid\_config\_loc, 62
  - support\_mail\_address, 62
- gridftpd::LdapQuery, 88
  - ~LdapQuery, 88
  - Host, 88
  - LdapQuery, 88
  - Query, 88
  - Result, 89
  - Scope, 88
- gridftpd::LdapQueryError, 90
  - LdapQueryError, 90
- gridftpd::ParallelLdapQueries, 98
- gridftpd::RunPlugin, 108
- gridftpd::RunPlugin::lib\_plugin\_t, 92
- gridftpd::UrlMapConfig, 121
- group
  - voms\_attrs, 125
- hasJob
  - ARex::DTRGenerator, 44
- Host
  - gridftpd::LdapQuery, 88
- ID
  - ARex::ARexJob, 13
- Identity, 67
- Identity::Item, 73
- IdentityGACL, 68
- IdentityItemDN, 69
- IdentityItemVOMS, 70
- Index, 71
- JobLogFile
  - Arc::JobLogFile, 83
- JobPlugin, 85
- Jobs
  - ARex::ARexJob, 14
- LdapQuery
  - gridftpd::LdapQuery, 88
- LdapQueryError
  - gridftpd::LdapQueryError, 90
- Load
  - ARex::GMConfig, 61
- LogDir
  - ARex::ARexJob, 14
- LogFiles
  - ARex::ARexJob, 14

- LutsDestination
  - Arc::LutsDestination, 94
- Modified
  - ARex::ARexJob, 14
- nordugrid\_config\_loc
  - gridftpd::GMEEnvironment, 62
- ObjectAccess, 95
- ObjectAccess::Item, 72
- ObjectAccessGACL, 96
- olderThan
  - Arc::JobLogFile, 84
- OpenDir
  - ARex::ARexJob, 14
- OpenFile
  - ARex::ARexJob, 14
- OpenLogFile
  - ARex::ARexJob, 14
- operator bool
  - Cache::CacheService, 23
- parse
  - Arc::JobLogFile, 84
- parse\_job\_req
  - ARex::JobDescriptionHandler, 76
- PayloadBigFile
  - ARex::PayloadBigFile, 99
- PayloadFAFile
  - ARex::PayloadFAFile, 100
- PayloadFile
  - ARex::PayloadFile, 101
- Permission, 102
- PermissionGACL, 103
- Policy, 104
- Process
  - ARex::DelegationStores, 36
- process
  - Arc::Service\_JavaWrapper, 113
  - Arc::Service\_PythonWrapper, 114
  - Cache::CacheService, 23
  - SPService::Service\_SP, 116
- Query
  - gridftpd::LdapQuery, 88
- queryJobFinished
  - ARex::DTRGenerator, 44
- queryRequestsFinished
  - Cache::CacheServiceGenerator, 24
- receiveDTR
  - ARex::DTRGenerator, 44
- receiveJob
  - ARex::DTRGenerator, 44
- RegistrationCollector
  - Cache::CacheService, 23
- Release
  - ARex::FileChunks, 49
- Remove
  - ARex::FileChunks, 49
- remove
  - Arc::JobLogFile, 84
- removeJob
  - ARex::DTRGenerator, 45
- report
  - Arc::ApelDestination, 10
  - Arc::CARDestination, 26
  - Arc::Destination, 37
  - Arc::Destinations, 38
  - Arc::LutsDestination, 94
  - Arc::UsageReporter, 122
- Result
  - gridftpd::LdapQuery, 89
- Resume
  - ARex::ARexJob, 14
- role
  - voms\_attrs, 125
- Scope
  - gridftpd::LdapQuery, 88
- Server, 110
- server
  - voms, 124
- Service\_SP
  - SPService::Service\_SP, 116
- SessionDir
  - ARex::ARexJob, 14
- SessionRoot
  - ARex::GMConfig, 61
- SimpleMap, 117
- SPService::Service\_SP, 116
  - process, 116
  - Service\_SP, 116
- StagingConfig
  - ARex::StagingConfig, 118
- State
  - ARex::ARexJob, 14
- Substitute
  - ARex::GMConfig, 61
- support\_mail\_address
  - gridftpd::GMEEnvironment, 62
- TotalJobs
  - ARex::ARexJob, 15
- UnixMap, 119
- UpdateCredentials
  - ARex::ARexJob, 15

---

UsageReporter  
    Arc::UsageReporter, [122](#)  
userspec\_t, [123](#)  
  
voms, [124](#)  
    attrs, [124](#)  
    server, [124](#)  
    voname, [124](#)  
voms\_attrs, [125](#)  
    cap, [125](#)  
    group, [125](#)  
    role, [125](#)  
voname  
    voms, [124](#)