

# EGEE

## Apel User Guide

---

Document identifier:	EGEE-JRA1-TEC
Date:	August 4, 2005
Activity:	JRA1: Middleware Engineering and Integration (UK Cluster)
Document status:	RELEASED
Document link:	

---

Abstract: This document describes how to setup your site to be part of the Apel accounting system. It describes how to install and configure the Log Parser and Publisher components.

### Document Change Log

Issue	Date	Comment	Author
1.0	26/04/05	First Draft	Rob Byrom, Dave Kant

### Document Change Record

Issue	Item	Reason for Change
-------	------	-------------------

Copyright © Members of the EGEE Collaboration. 2004. See <http://eu-egEE.org/partners> for details on the copyright holders.

EGEE (“Enabling Grids for E-science in Europe”) is a project funded by the European Union. For more information on the project, its partners and contributors please see <http://www.eu-egEE.org>.

You are permitted to copy and distribute verbatim copies of this document containing this copyright notice, but modifying this document is not allowed. You are permitted to copy this document in whole or in part into other documents if you attach the following reference to the copied elements: “Copyright © 2004. Members of the EGEE Collaboration. <http://www.eu-egEE.org>”

The information contained in this document represents the views of EGEE as of the date they are published. EGEE does not guarantee that any information contained herein is error-free, or up to date.

EGEE MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, BY PUBLISHING THIS DOCUMENT.

## CONTENTS

1	DISCLAIMER	4
2	INTRODUCTION	5
2.1	WHERE R-GMA FITS IN	5
2.2	R-GMA SERVER GUIDELINES	5
3	APEL CORE	7
3.1	DEPLOYMENT	7
3.2	INSTALLATION	7
4	LOG PARSER	8
4.1	DEPLOYMENT	8
4.1.1	SEPARATE GK/CE	8
4.1.2	COMBINED GK/CE	8
4.2	INSTALLATION	8
4.2.1	LOG PARSER COMPONENTS	9
4.3	CONFIGURATION	9
4.3.1	PREAMBLE	10
4.3.2	DBDELETEPROCESSOR	10
4.3.3	CPUPROCESSOR	11
4.3.4	EVENTLOGPROCESSOR	11
4.3.5	GKLOGPROCESSOR	12
5	LOG PARSER EXAMPLES	13
5.1	SEPARATE GK/CE	13
5.1.1	GK CONFIGURATION	13
5.1.2	CE CONFIGURATION	13
5.2	COMBINED GK/CE	14
5.2.1	GK/CE CONFIGURATION	14
6	PUBLISHER	16
6.1	DEPLOYMENT	16
6.2	INSTALLATION	16
6.2.1	PUBLISHER COMPONENTS	16
6.3	CONFIGURATION	16
6.3.1	JOINPROCESSOR	16
7	PUBLISHER EXAMPLES	18
7.1	R-GMA SERVER CONFIGURATION	18

## 1 DISCLAIMER

This guide provides an overview of the Apel accounting software, where to install it and how to go about installing it. It does not provide an installation guide for setting up an R-GMA system. Users wishing to find out more about how to do this should refer to documentation available on the [R-GMA web site](http://hepunix.rl.ac.uk/egEE/jra1-uk/)<sup>1</sup>.

---

<sup>1</sup><http://hepunix.rl.ac.uk/egEE/jra1-uk/>

## 2 INTRODUCTION

The Apel software is composed of two components: The Log Parser and Publisher.

The Log Parser interprets log files to extract job information and publishes it using R-GMA. Specifically, it processes the LCG gatekeeper logs, the system message logs and PBS or LSF event logs. Extracted data is then stored within a MySQL database. The Apel Log Parser also makes LDAP queries of the Computing Element to obtain the CPU performance figures for the worker node clusters and sub-clusters.

The Publisher is used to generate accounting records derived from the parsed logging data. The accounting records are then published into R-GMA where they are then collected by a central accounting server which, aggregates records from all sites.

### 2.1 WHERE R-GMA FITS IN

The Apel software uses R-GMA as a transport mechanism for moving accounting records stored per site to a centralised database. This is achieved by using a primary producer to publish accounting records (performed by the Publisher) and a secondary producer which is setup on the centralised store (see Figure 1).

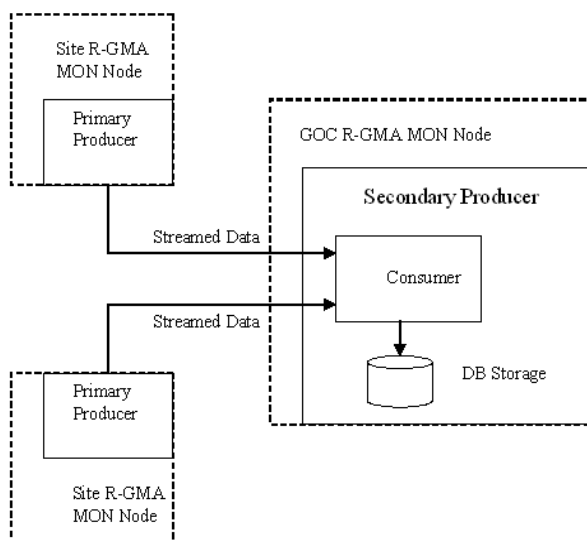


Figure 1: Accounting data is streamed from each site to a secondary producer

For the purpose of this user guide, we assume the secondary producer already exists (i.e. it's been configured by a sys admin located at the central accounting site).

For a more detailed overview of R-GMA, an excellent introduction is covered by the [R-GMA specification](#)<sup>2</sup>.

### 2.2 R-GMA SERVER GUIDELINES

The primary producer created by the Publisher is managed by an R-GMA server located locally on the same site. We recommend each site sets up its own R-GMA server, the reasons for this are summarised:

accounting data may hold sensitive data which is private to each site.

<sup>2</sup><https://edms.cern.ch/document/490223/2>

the site administrator has direct control over the data and can control when locally stored accounting records can be deleted, regenerated and so on.

the exact number of sites an R-GMA server can support is unclear.

For optimal performance, we recommend installing the R-GMA server on a dedicated host. Ideally this should be separate from the CE and GateKeeper hosts.

### 3 APEL CORE

The Apel Core contains common functionality used by the Log Parser and Publisher components, it does not provide any direct functionality to the user.

#### 3.1 DEPLOYMENT

Since the Log Parser and Publisher depend upon an Apel Core, this module must be deployed on all hosts running either component.

#### 3.2 INSTALLATION

The Apel Core comes in rpm format available from [EGEE web site](http://glite.web.cern.ch/glite/packages/)<sup>3</sup>. The package contains a number of files, the following lists those the sys admin should pay attention to:

`/opt/glite/share/glite-apel-core/scripts/apel-db-config.py` this script is used to setup a MySQL database for a particular user. It will also install the schema used by both the Log Parser and Publisher. Make sure you run this script where your MySQL server is located.

On each host where the Apel Core is installed, make sure the following environment variable has been set:

```
export APEL_HOME=/opt/glite
```

When installing the Apel Core, the `apel-db-config.py` script must be run on the same host as the MySQL server (this will be the same host as the R-GMA server). Keep a note of the name of the database, user and password you use when running the script (you will need these when configuring the Log Parser and Publisher).

---

<sup>3</sup><http://glite.web.cern.ch/glite/packages/>

## 4 LOG PARSER

The Log Parser is used to parse gate keeper, system message and event logs produced by a site running a batch processing system. Currently, the Log Parser comes in two different flavours: PBS and LSF. The Log Parser you choose will depend upon the batch processing system used by the site. However, the deployment and configuration are both equivalent.

### 4.1 DEPLOYMENT

The Log Parser can be deployed in a variety of ways depending upon the setup of your site. An overview of some of the typical deployments are described as follows.

#### 4.1.1 SEPARATE GK/CE

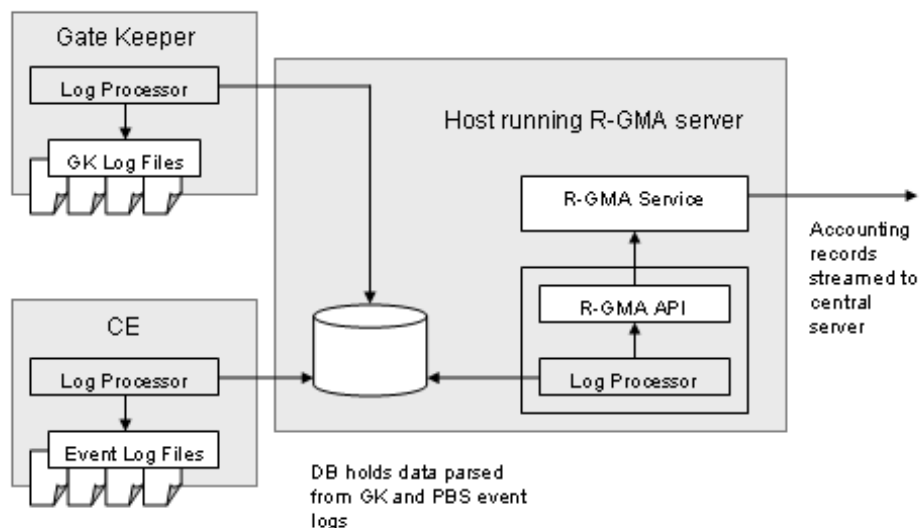


Figure 2: Shows Gate Keeper and CE on different hosts

See Fig2. In this example, the gate keeper and CE are hosted on separate machines; both require an installation of the Log Parser. On the gate keeper, the Log Parser is configured to process gate keeper logs. Whereas the configuration on the CE is tailored to cater for event logs (conforming to LSF or PBS format).

Fig 2 also shows the R-GMA server. This is where the Publisher is normally installed (described in

#### 4.1.2 COMBINED GK/CE

See Fig3. In this setup the Log Parser is configured to process both the gate keeper logs and event logs but on the same host.

### 4.2 INSTALLATION

The Log Parser comes with its own rpm package available from [EGEE web site](http://glite.web.cern.ch/glite/packages/)<sup>4</sup>. It requires the Apel Core module to be installed (available from the same repository).

<sup>4</sup><http://glite.web.cern.ch/glite/packages/>



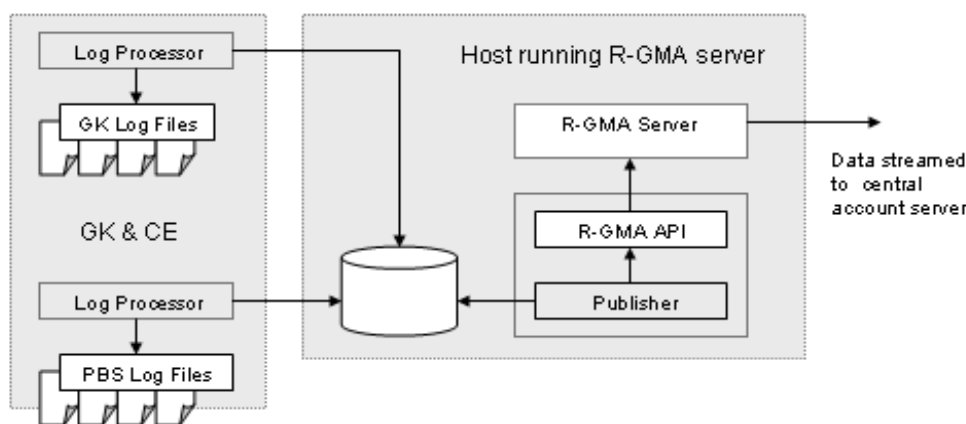


Figure 3: Shows Gate Keeper and CE combined on the same host

#### 4.2.1 LOG PARSER COMPONENTS

The rpm installs the following components (using the PBS Log Parser as an example):

`/opt/glite/bin/apel-pbs-log-parser` the script used to run the Log Parser.  
`/opt/glite/etc/glite-apel-pbs/parser-config.xml` contains an example configuration file used by the Log Parser.

The same files can be found when installing the LSF Log Parser by substituting 'pbs' with 'lsf'.

The Log Parser should be run daily. To schedule the Log Parser to run at 2:20 am every day, setup a cron entry as follows:

```
20 2 * * * /opt/glite/bin/apel-pbs-log-parser -f
/opt/glite/etc/glite-apel-pbs/config.xml > /var/log/apel.log 2>&1
```

In addition, it is a good idea to define a log rotation script, an example is shown as follows:

```
/var/log/apel.log {
    copy
}
```

The log rotation script in this example is stored under `/etc/logrotate.d/apel`. Note, the cron and log rotate examples may need some tweaking depending upon the flavour of linux you use.

The sys admin should periodically backup data stored in the MySQL database used by the Log Parser. If tables grow large enough to cause disk space shortage, the sys admin is advised to back-up the database (using `mysqldump`) before purging the database contents.

#### 4.3 CONFIGURATION

The Log Parser is setup using a configuration file encoded in an XML format. The configuration file is composed of a list of processors. Each processor carries out a unit of functionality.

When the Log Parser is started, the program will find any processors defined within the configuration file and will attempt to schedule each one consecutively. The list of configurable processors in order of execution are:

DBDeleteProcessor

CPUProcessor

EventLogProcessor

GKLogProcessor

An overview of the configuration format is detailed as follows.

#### 4.3.1 PREAMBLE

The configuration file contains a preamble containing a number of fields that must be filled in by the user. The most important are the database credentials i.e. the database url, username and password. These credentials define the MySQL database to be used when storing parsed data derived from the gate keeper, system message and event log files.

An example of the preamble is shown as follows:

```

<?xml version="1.0" encoding="UTF-8"?>
<ApelConfiguration enableDebugLogging="yes">
  <DBURL>jdbc:mysql://grid-example.xy.ac.uk:3306/lcg</DBURL>
  <DBUsername>lcg</DBUsername>
  <DBPassword>no default</DBPassword>
  <SiteName>RAL-LCG2</SiteName>
</ApelConfiguration>
  
```

enableDebugLogging enables the user logging.

DBURL the URL that points to the MySQL database where the log data is to be recorded. Replace the hostname part with the hostname where your database is located.

DBUsername/DBPassword the username and password to be used to access the database on the MySQL database. Since a password is stored in this file you should setup suitable file access privileges.

SiteName is the name of the site producing the accounting data. This must be set to the Sites' public Distinguished Name (DN) and must be consistent in all Apel configuration files used by the site.

#### 4.3.2 DBDELETEPROCESSOR

The DBDeleteProcessor is used to delete records from the accounting db. An example of the delete processor is shown as follows:

```

<DBDeleteProcessor cleanAll="yes"/>
  
```

Special attention should be taken when using such a configuration. As a general rule, separate the delete processor within its own configuration file to avoid any unwanted deletion while parsing new log files.

CleanAll : If enabled, all table contents in the accounting database will be purged. If disabled, only records that were successfully merged to produce accounting records are removed (this will only delete gate keeper, system message and event log file data stored in the database).

#### 4.3.3 CPUPROCESSOR

If this processor is present, CPU performance information will be generated and inserted into the database to be used when publishing accounting records. Normally this information comes from an LDAP query to a GIIS server, specified by the `<GIIS>` element. However if this is not available it is possible to set a default value using the `<DefaultCPUSpec>` element using the following format: `intSpec:floatSpec`.

An example using a GIIS host.

```
<CPUProcessor>
  <GIIS>lcgce.example</GIIS>
</CPUProcessor>
```

An example that uses a default spec value instead:

```
<CPUProcessor>
  <DefaultCPUSpec>300:200</DefaultCPUSpec>
</CPUProcessor>
```

This processor should be run with the `EventLogProcessor` (see 4.3.4). It is crucial that spec data is available when accounting records are generated by the Publisher.

Note, if the GIIS is unavailable, the Log Parser will log the error and will terminate. If this occurs, the admin user should re-run the script using an estimated default spec value (or investigate the health of the GIIS).

#### 4.3.4 EVENTLOGPROCESSOR

The `EventLogProcessor` is used to parse event logs. The type of event logs parsed depend upon the log parser installed (eg PBS or LSF). This processor is usually run on the CE where the event logs are generated. An example config is shown as follows:

```
<EventLogProcessor>
  <Logs searchSubDirs="yes" reprocess="no">
    <Dir>/var/spool/pbs/server_priv/accounting</Dir>
    <ExtraFile>/home/toucan/pbslog00021123.gz</ExtraFile>
    <ExtraFile>/home/toucan/pbslog00021100.log</ExtraFile>
  </Logs>
  <Timezone>UTC</Timezone>
</EventLogProcessor>
```

`Dir` : contains the directory where the event logs are to be found.

`ExtraFile` : an optional parameter that allows additional filenames to be added to the list of event log files to be processed.

`SearchSubDirs`: if enabled, all sub directories will be searched for event logs.

`Reprocess` when an event log file is processed, the log file name is stored in the database to prevent the file from being reprocessed in future. Enabling this option will force all event files to be re-parsed. Using this option is a safe way for re-building data from a batch of log files.

Timezone the event logs may be processed on a different machine from where they originated. For example, an admin user may want to run the Log Parser using event logs generated from a different site. Normally, the EventLogProcessor on each site will convert between UTC and local time of the processing host. However, this is not so useful where log files are being processed in a different timezone to that where they were generated.

The behaviour can be modified using the following options:

- derived : looks at the start and end time stamp of each job in the PBS event log (in UTC) to derive the timezone when the record was written.
- default : uses the host's timezone.
- any other value : for example 'GMT+1:30' indicates that 1 hour and 30 minutes has to be added to UTC to obtain the local time where the event log file was generated.

#### 4.3.5 GKLOGPROCESSOR

The GKLogProcessor is used to process the gate keeper and system message logs generated on the Gate Keeper host. An example with an explanation of each field is shown as follows:

```
<GKLogProcessor>
  <SubmitHost>lcgce.example</SubmitHost>
  <Logs searchSubDirs="yes" reprocess="no">
    <GKLogs>
      <Dir>/var/log</Dir>
    </GKLogs>
    <MessageLogs>
      <Dir>/var/log</Dir>
    </MessageLogs>
  </Logs>
</GKLogProcessor>
```

SubmitHost: hostname of the gate keeper (ie the entry point in the site where jobs are submitted).

GKLogs contains the directory where the gate keeper log files will be found. gate keeper log file names contain a date suffix appended to globus-gatekeeper and may be of gzip format. File names examples are shown as follows:

```
globus-gatekeeper.log
globus-gatekeeper.log.20040125040202.0
globus-gatekeeper.log.20040201040203.0.gz
```

MessageLogs contains the directory where the system message log files are found. As with other files, these are recognised by having a particular filename format, examples of which are shown as follows:

```
messages.1.gz
messages.2.gz
messages.200401201123904.gz
messages.3.gz etc
```

SearchSubDirs: if enabled all sub directories will be searched for gate keeper logs.

Reprocess: Same as the attribute found in EventLogProcessor.

## 5 LOG PARSER EXAMPLES

### 5.1 SEPARATE GK/CE

This example is based on the setup described in [4.1.1](#). The CE and Gate Keeper are situated on different machines each running the Log Parser. The configuration used for each Log Processor is described as follows:

GK will run the GKLogProcessor

CE will run the EventLogProcessor and CPUProcessor

#### 5.1.1 GK CONFIGURATION

```

<?xml version="1.0" encoding="UTF-8"?>
<ApelConfiguration enableDebugLogging="yes">
  <DBURL>jdbc:mysql://grid-example.xy.ac.uk:3306/lcg</DBURL>
  <DBUsername>test</DBUsername>
  <DBPassword>info</DBPassword>
  <SiteName>gridka.de</SiteName>
  <GKLogProcessor>
    <SubmitHost>hik-lcg-ce.fzk.de</SubmitHost>
    <Logs searchSubDirs="yes" reprocess="no">
      <GKLogs>
        <Dir>/var/data/gk</Dir>
      </GKLogs>
      <MessageLogs>
        <Dir>/var/message-data</Dir>
      </MessageLogs>
    </Logs>
  </GKLogProcessor>
</ApelConfiguration>

```

The Gate Keeper configuration only uses the GKLogProcessor. This processor will recursively search all directories below /var/data/gk and /var/message-data in-order to find any relevant log files. Since the 'reprocess' attribute is disabled, only new files will be processed.

Note, the DB credentials given are those used to access the DB located on the R-GMA server (which contains an installation of the MySQL server).

#### 5.1.2 CE CONFIGURATION

```

<?xml version="1.0" encoding="UTF-8"?>
<ApelConfiguration enableDebugLogging="yes">
  <DBURL>jdbc:mysql://grid-example.xy.ac.uk:3306/lcg</DBURL>
  <DBUsername>test</DBUsername>
  <DBPassword>info</DBPassword>
  <SiteName>gridka.de</SiteName>
  <CPUProcessor>
    <GIIS>pbs-server2.gridka.de</GIIS>
  </CPUProcessor>

```

```
<EventLogProcessor>
  <Logs searchSubDirs="yes" reprocess="no">
    <Dir>/var/data/el</Dir>
  </Logs>
  <Timezone>UTC</Timezone>
</EventLogProcessor>
</ApelConfiguration>
```

When the Log Parser is started, the CPUProcessor will be invoked first. It will try to query the GLIS (pbs-server2.gridka.de) in-order to retrieve a benchmark value of the site's CPU resources. Then all event logs found under /var/data/el will be processed (the search will include all sub directories).

## 5.2 COMBINED GK/CE

In this example, a site contains a GK and CE that, share the same host (based on the setup described in 4.1.2).

### 5.2.1 GK/CE CONFIGURATION

```
<?xml version="1.0" encoding="UTF-8"?>
<ApelConfiguration enableDebugLogging="yes">

  <DBURL>jdbc:mysql://grid-example.xy.ac.uk:3306/lcg</DBURL>
  <DBUsername>test</DBUsername>
  <DBPassword>info</DBPassword>
  <SiteName>RAL-LCG2</SiteName>

  <CPUProcessor>
    <GLIS>lcgce.example</GLIS>
  </CPUProcessor>

  <EventLogProcessor>
    <Logs searchSubDirs="yes" reprocess="yes">
      <Dir>/var/spool/pbs/server_priv/accounting</Dir>
      <ExtraFile>/home/toucan/pbslog00021123.gz</ExtraFile>
      <ExtraFile>/home/toucan/pbslog00021100.log</ExtraFile>
    </Logs>
    <Timezone>UTC</Timezone>
  </EventLogProcessor>

  <GKLogProcessor>
    <SubmitHost>lcgce.example</SubmitHost>
    <Logs searchSubDirs="yes" reprocess="yes">
      <GKLogs>
        <Dir>/var/log</Dir>
      </GKLogs>
      <MessageLogs>
        <Dir>/var/log</Dir>
      </MessageLogs>
    </Logs>
```

</GKLogProcessor>

</ApelConfiguration>

In this setup, the CPUProcessors run first and is configured to query the GLIS server named lcgce.example. Then EventLogProcessor is will look for all log files located under the Dir path, it will also attempt to process the given extra files. The GKLogProcessor is the last to be scheduled and will process all GK logs. In this example all log files found will be re-parsed.

Note, in both the GK and CE processors, the searchSubDirs option is enabled. This is the recommended option to ensure all files located under the given Dir are processed.

## 6 PUBLISHER

The Publisher is used to piece together accounting records derived from data parsed from gate keeper, system message and event log files. The generated data is stored locally within a MySQL database and is also published into R-GMA. The published data will then be stored by R-GMA onto the central accounting server.

### 6.1 DEPLOYMENT

The Publisher is deployed on the same host that runs the MySQL server. By default, the R-GMA server comes equipped with a MySQL server installation so we recommended this as the best place to deploy the Publisher.

### 6.2 INSTALLATION

The Publisher comes with its own rpm package available from [the EGEE web site](#)<sup>5</sup>. It requires the Apel Core rpm to be installed (also available from the same repository).

The Publisher also requires an R-GMA client installation, please refer to installation notes available on the [R-GMA web site](#)<sup>6</sup>.

#### 6.2.1 PUBLISHER COMPONENTS

The rpm installs the following components:

/opt/glite/bin/apel-publisher	Script used to run the Publisher.
/opt/glite/etc/glite-apel-publisher/publisher-config.xml	Contains an example configuration file used by the Publisher.

Similar to the Log Parser, the publisher should be scheduled to run on a daily basis. See notes described in [4.2.1](#).

### 6.3 CONFIGURATION

The Publisher is setup using a configuration file encoded in an XML format. Compared to the Log Parser, the configuration file is composed of only two processors: `DBDeleteProcessor` and `JoinProcessor`.

The configuration format uses the same header as that used by the Log Parser [4.3.1](#). The `DBDeleteProcessor` is the same as that defined in [4.3.2](#).

#### 6.3.1 JOINPROCESSOR

The `JoinProcessor` is run on the R-GMA server. It attempts to join data stored by the Log Parser to build accounting records. An example configuration is shown as follows:

```
<JoinProcessor publishGlobalUserName="no">
  <Republish>all</Republish>
</JoinProcessor>
```

<sup>5</sup><http://glite.web.cern.ch/glite/packages/>

<sup>6</sup><http://hepunix.rl.ac.uk/egEE/jra1-uk/>



publishGlobalUserName: if disabled, the Distinguished Name (DN) of the user in each accounting record is set to 'NULL' when published into R-GMA. Some sites may wish to suppress the DN for reasons of personal privacy. By default, the DN will always be suppressed from publication.

Republish: Contains a list of options the user can choose when republishing accounting data. These are listed as follows:

- all : setting the republish option to 'all' causes all accounting data in the database to be regenerated and published. This option should never be set as a long term configuration. Doing so will impair the performance of the R-GMA server especially if the volume of accounting records is huge (order of tens of thousands)
- missing : in some cases, the central accounting server may not receive the accounting records that were published in a previous join. Rather than having to republish the complete set, the JoinProcessor can send a subset of data beginning from the last successful transfer. So when the JoinProcessor runs, a timestamp is recorded and the data is published as normal. When the JoinProcessor is next invoked, a count of all records that are greater than or equal to the timestamp is derived from the central accounting server. This is compared with the local database and if the accounting server contains fewer records, the JoinProcessor will republish all records beginning from the timestamp. When the accounting server and local databases are in synch, the timestamp will then be updated as normal. This is the default behaviour.
- nothing : disables the republishing mode.

## 7 PUBLISHER EXAMPLES

The Publisher configuration remains unchanged irrespective of where the CE or GK is setup (see [4.1.1](#) and [4.1.2](#)). This is because the Publisher is deployed on the R-GMA server.

### 7.1 R-GMA SERVER CONFIGURATION

An example Publisher configuration used on the R-GMA server is shown as follows:

```

<?xml version="1.0" encoding="UTF-8"?>
<ApelConfiguration enableDebugLogging="yes">
  <DBURL>jdbc:mysql://grid-example.xy.ac.uk:3306/lcg</DBURL>
  <DBUsername>gridka-mon</DBUsername>
  <DBPassword>mumbojumbo</DBPassword>
  <SiteName>gridka.de</SiteName>
  <JoinProcessor publishGlobalUserName="no">
    <Republish>missing</Republish>
  </JoinProcessor>
</ApelConfiguration>
  
```

In this example, the `JoinProcessor` is setup to connect to the same MySQL database as the one used by the GK and CE. This is where the log data is held so the credentials of the DB are defined in this configuration as well.

The `JoinProcessor` will not reprocess all log data when generating the accounting records - instead it will concentrate on new data only. However, it will carry out a check to make sure all tuples sent in the last join were successfully received by the central accounting server. If the tuples have not been received it will then attempt to republish the set of records from the last successful synchronization check.

## REFERENCES