Steve Crouch, OMII-UK, University of Southampton
Donal Fellows, University of Manchester

March 23, 2009
Updated: March 30, 2009

Experiences of Using Usage Record (UR) version 1.0

Status of This Document

This document provides information to the Grid community on experiences of using the Usage Record (UR) format recommendation version 1.0. It does not define any standards or technical recommendations. Distribution is unlimited.

Abstract

For resources to be shared, sites must be able to exchange basic accounting and usage data in a common format. This format must encompass both job level accounting and aggregate accounting. The Usage Record (UR) Format describes a common format with which to exchange basic accounting and usage data over a grid instantiation.

This document describes experiences of using the Usage Record (UR) Format version 1.0 Recommendation across a number of infrastructures.

Contents

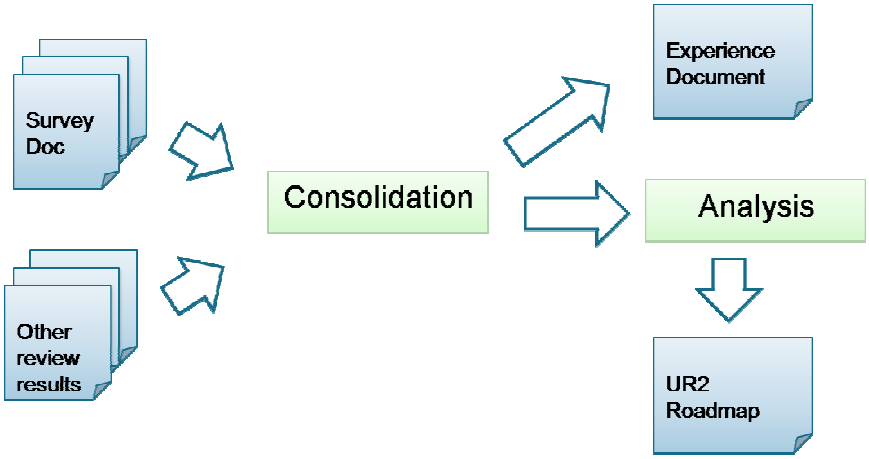
1. Introduction

For resources to be shared, sites must be able to exchange basic accounting and usage data in a common format. This format must encompass both job level accounting and aggregate accounting. The Usage Record (UR) Format describes a common format with which to exchange basic accounting and usage data over a grid instantiation.

This document describes experiences of using the Usage Record (UR) Format version 1.0 Recommendation across a number of infrastructures. The document is based on information returned from the experiences survey activity.

2. Process & Objectives

The process of gathering and consolidating the experiences is as follows:



The artifacts and activities in the above process correspond to the following objectives:

- **Survey doc/Other review results:** from the results of the survey activity, and other sources, gather:
  - Use case collection – information concerning how UR1.0 is used.
  - Implementation status – the status of each of these infrastructures in their adoption of UR1.0
- **Consolidation:** from this gathered information, consolidate advanced requirements that have emerged into the *Experiences Document*.
- **Experience Document [this document]:**
  - Bring together the results from the consolidation activity
  - Provide guidance for how to progress towards UR2
  - Provide a reflective role for the RUS-WG activities.

Additionally, there is a parallel **Analysis** objective to produce an **UR2 Roadmap** from the consolidation activity and experiences. However, this is out of scope of this document [or is it?]

3. Survey Template

The survey template used to gather experiences with UR1.0 (available at http://forge.ogf.org/sf/go/artf6130?nav=1) was comprised of six parts:

1. **Project Description:** overview of the project making use of UR1.0.
2. **Usage Metric Checklist:** indication of which elements and attributes of the UR1.0 specification are produced and/or consumed.
3. **Extensions (if any):** extensions to the specification that have been adopted to meet with additional requirements, and an indication of which extensions are considered important.
4. **Metering time:** the times that usage records are generated e.g. at job start, job end, at points during the job, etc.
5. **High-level usage:** details of what is done with usage records after generation.
6. **Recommendation to UR2 Roadmap:** in which directions should the Usage Record evolve? e.g. summary records, aggregate records, improved execution/data/network accounting, etc.
7. **Feedback:** opinions on any missing questions that should have been included in the survey.

4. Identified Use Cases

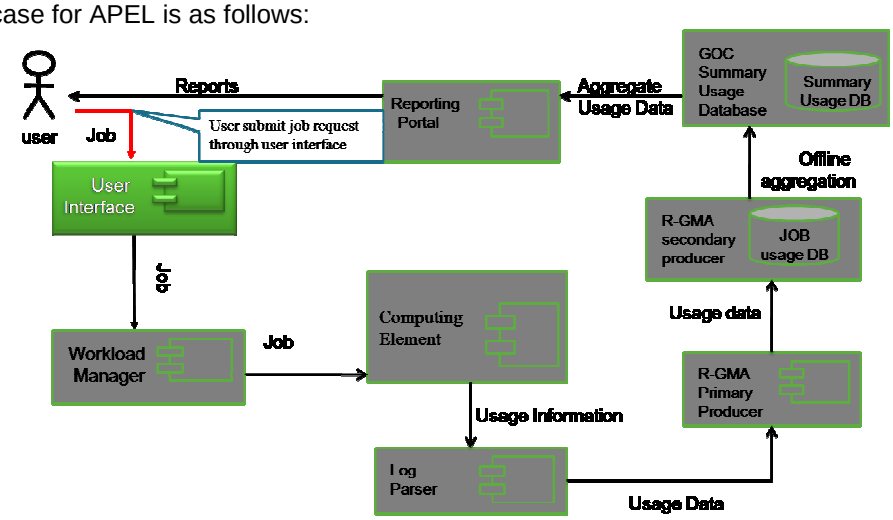
4.1 APEL

4.1.1 Overview

APEL (Accounting Processor for Event Logs), is the accounting system of WLCG, which uses URs for reporting usage statistics. Its key components are the *Log Parser*, which collates the information to create the records, *R-GMA* as the publishing protocol, and the *Reporting Portal*.

4.1.2 APEL Use Case

The use case for APEL is as follows:



1. User submits job request through the *User Interface*.
2. The request is forwarded to the *Workload Manager* for scheduling.
3. The job is scheduled to a particular *Computing Element* for execution.

4. The APEL *Log Parser* is deployed at the *Computing Element* site to meter usage information periodically (e.g. once per day) by parsing LRMS log files and other relevant services.
5. Pieces of usage information are gathered as a complete usage record and put into an *R-GMA* virtual database as a *primary producer*.
6. Usage data are shipped into GOC/CCLRC's persistent *job usage database*.
7. An *offline aggregation* process usage records on per site, per VO, per month, per year basis into the *GOC Summary Usage Database*.
8. User queries usage statistics through the *Reporting Portal*.

4.1.3 APEL Schema and UR

The Usage Schema adopted by APEL is as follows:

APEL Usage Property		OGF UR Metric	
Name	**Data Type (SQL)**	**Context Node**	**Data Type (XSD)**
RecordIdentity	VARCHAR	/urf:UsageRecord/urf:RecordIdentity/@recordId	xsd:string
ExecutingSite	VARCHAR	/urf:UsageRecord/urf:Resource[@description="ExecutingSite"]	xsd:string
LCGJobId	VARCHAR	/urf:UsageRecord/urf:JobIdentity/urf:GlobalJobId	xsd:string
LocalJobId	VARCHAR	/urf:UsageRecord/urf:JobIdentity/urf:LocalJobId	xsd:string
LCGUserId	VARCHAR	/urf:UsageRecord/urf:UserIdentity/urf:GlobalUserName	xsd:string
LocalUserId	VARCHAR	/urf:UsageRecord/urf:UserIdnetity/urf:LocalUserId	xsd:string
LCGUserVO	VARCHAR	/urf:UsageRecord/urf:Resource[@description="LCGUserVO"]	xsd:string
ElapsedTime	VARCHAR	/urf:UsageRecord/urf:WallDuration	xsd:duration
BaseCpuTime	VARCHAR	/urf:UsageRecord/urf:CpuDuration	xsd:duration
ElapsedTimeSeconds	INTEGER	/urf:UsageRecord/urf:ConsumableResource[@urf:description="ElaspedTimeInSeconds"]	xsd:string
BaseCpuTimeSeconds	INTEGER	/urf:Usagerecord/urf:ConsumableResecource[@urf:description="BaseCpuTimeSeconds"]	xsd:string
StopTime	VARCHAR	/urf:UsageRecord/urf:EndTime	xsd:dateTime
StartTime	VARCHAR	/urf:UsageRecord/urf:StartTime	xsd:dateTime
StopTimeUTC	VARCHAR	/urf:UsageRecord/urf:EndTime[@urf:description="EndTimeUTC"]	xsd:dateTime
StartTimeUTC	VARCHAR	/urf:UsageRecord/urf:StartTime[@urf:description="StartTimeUTC"]	xsd:dateTime
StopTimeEpoch	INTEGER	/urf:UsageRecord/urf:Resource[@urf:description="EndTimeEpoch"]	xsd:string
StartTimeEpoch	INTECER	/urf:UsageRecord/urf:Resource[@urf:description="StartTimeEpoch"]	xsd:string
ExecutingCE	VARCHAR	/urf:UsageRecord/urf:Resource[@urf:description="ExecutingCE"]	xsd:string
MemoryReal	INTEGER	/urf:UsageRecord/urf:Memory[@urf:storageUnit="KB"][@urf:type="real"]	xsd:int
MemoryVirtual	INTEGER	/urf:UsageRecord/urf:Memory[@urf:storagetUnit="KB"][@urf:type="virtual"]	xsd:int
SpecInt2000	INTEGER	/urf:UsageRecord/urf:Resource[@urf:description="SpecInt2000"]	xsd:string
SpecFloat2000	INTEGER	/urf:UsageRecord/urf:Resource[@urf:description="SpecFloat2000"]	xsd:string
EventDate	Date	/urf:UsageRecord/urf:TimeInstant[@urf:description="MeasurementDateTime"]	xsd:dateTime
EventTime	Time	/urf:UsageRecord/urf:TimeInstant[@urf:description="MeasurementDateTime"]	xsd:dateTime
MeasurementDate	Date	/urf:UsageRecord/urf:RecordIdentity/@urf:CreateTime	xsd:dateTime
MeasurementTime	Time	/urf:UsageRecord/urf:RecordIdentity/@urf:CreateTime	xsd:dateTime

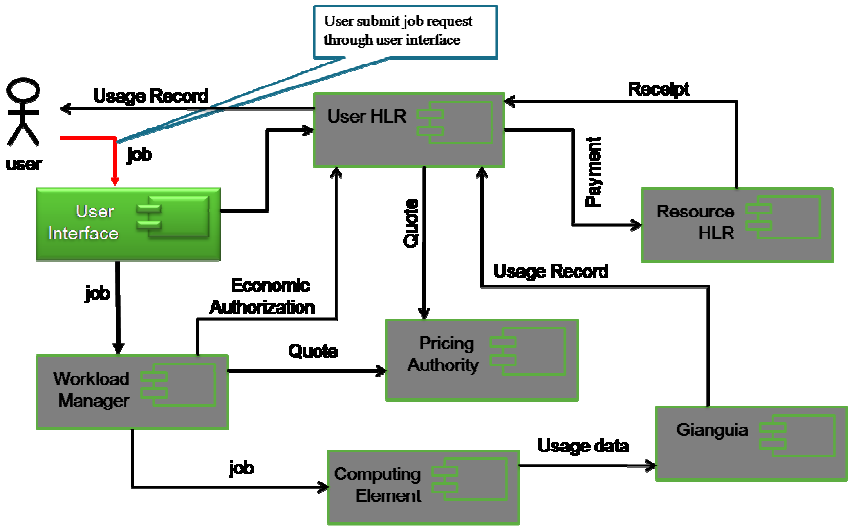
4.2 DGAS

4.2.1 Overview

DGAS (Distributed Grid Accounting System) employed by EGEE, used for economic accounting (e.g. economic authorization, pricing, banking, etc). Its key components are the *User Interface (UI)*, the *Workload Manager*, the *Price Authority*, the *Home Location Registry (HLR)* and *Sensor*.

4.2.2 DGAS Use Case

The use case for DGAS is as follows:



1. User submit job request through the *User Interface (UI)*.
2. The *User Home Location Registry (HLR)* maintains user accounts. The user interface contacts *User HLR* to ensure the user has an account.
3. Once authorized, the job request is forwarded into the *Workload Manager* for scheduling.
4. The *Workload Manager* estimates the computation duration of the job request and quotes this estimation to the *Pricing Authority* for price estimation.
5. The *Workload Manager* then contacts the *User HLR* to execute economic authorization and check whether the user enough credits to run the job.
6. Once authorized, the job is scheduled or queued to a specific at *Computing Element*.
7. On successful completion of the job, the Metering Client [Gianguia] embedded within the *Computing Element* resource composes a usage record of the job and forwards it to the *User HLR*.
8. Gianguia has two client APIs, the ATM Client and URWGMetering API that feeds usage records into the *HLR* relational database and XML database respectively.
9. The *User HLR* sends the actual usage as specified within the received usage record of the job to the *Pricing Authority* to get the actual price.
10. The *User HLR* then contacts the *Resource HLR*, where user-resource accounts are maintained, and executed payment.
11. The *Resource HLR* sends back a receipt to the *User HLR* upon completion of accounting process.
12. User interacts with the *User HLR* APIs to query usage records.

4.2.3 DGAS Schema and UR

The Usage Schema of the DGAS ATM Client API is as follows:

DGAS Usage Property		UR Metric	
Name	Data Type (SQL)	Context Node	Data Type (XSD)
dgJobId	VARCHAR	/urf:UsageRecord/urf:RecordIdentity/@recordId	xsd:string
time	INTEGER	/urf:UsageRecord/urf:RecordIdentity/@createTime	xsd:dateTime
res_acct_PA_id	VARCHAR	/urf:UsageRecord/urf:Resource[@urf:description="res_acct_PA_id"]	xsd:string
res_acct_bank_id	VARCHAR	/urf:UsageRecord/urf:Resource[@urf:description="res_acct_bank_id"]	xsd:string
user_CertSubject	VARCHAR	/urf:UsageRecord/urf:UserIdentity/ds:KeyInfo/ds:X509Data/ds:X509SubjectName	xsd:string
res_grid_id	VARCHAR	/urf:UsageRecord/urf:Resource[@urf:description="res_grid_id"]	xsd:string
economicAccountingFlag	BOOLEAN	/urf:UsageRecord/urf:Resource[@urf:description="economicAccountingFlag"]	xsd:string
CPU_TIME	INTEGER	/urf:UsageRecord/urf:CpuDuration	xsd:duration
WALL_TIME	INTEGER	/urf:UsageRecord/urf:WallDuration	xsd:duration
PMEM	INTEGER	/urf:UsageRecord/urf:Memory[@storageUnit="KB"][@type="physical"]	xsd:int
VMEM	INTEGER	/urf:UsageRecord/urf:Memory[@storageUnit="KB"][@type="virtual"]	xsd:int
QUEUE	VARCHAR	/urf:UsageRecord/urf:Queue	xsd:string
USER	VARCHAR	/urf:UsageRecord/urf:UserIdentity/urf:LocalUserId	xsd:string
LRMSID	VARCHAR	/urf:UsageRecord/urf:Resource[@urf:description="LRMSID"]	xsd:string
PROCESSORS	INTEGER	/urf:UsageRecord/urf:Processors	xsd:int
URCREATION	DATETIME	/urf:UsageRecord/urf:TimeInstant[@urf:description="URCREATION"]	xsd:dateTime
group	VARCHAR	/urf:UsageRecord/urf:Resource[@urf:description="group"]	xsd:string
jobName	VARCHAR	/urf:UsageRecord/urf:JobName	xsd:string
start	INTEGER	/urf:UsageRecord/urf:StartTime	xsd:dateTime
end	INTEGER	/urf:UsageRecord/urf:EndTime	xsd:dateTime
exitStatus	INTEGER	/urf:UsageRecord/urf:Status	xsd:string
ctime	INTEGER	/urf:UsageRecord/urf:TimeInstant[@urf:description="ctime"]	xsd:dateTime
qtime	INTEGER	/urf:UsageRecord/urf:TimeInstant[@urf:description="qtime"]	xsd:dateTime
etime	INTEGER	/urf:UsageRecord/urf:TimeInstant[@urf:description="etime"]	xsd:dateTime
si2k	INTEGER	/urf:UsageRecord/urf:Resource[@urf:description="si2k"]	xsd:string
sf2k	INTEGER	/urf:UsageRecord/urf:Resource[@urf:description="sf2k"]	xsd:string
tz	VARCHAR	/urf:UsageRecord/urf:Resource[@urf:description="tz"]	xsd:string
fqan	VARCHAR	/urf:UsageRecord/urf:Resource[@urf:description="fqan"]	xsd:string

The DGAS UR-WG Metering Client generates OGF-UR (v1) compatible records in XML, with additional extensions that are economic activity specific, as follows:

Extension	Description	OGF UR Mapping
resAccountingPaID	Pricing Authority Identity	/urf:UsageRecord/urf:Resource[@urf:description="resAccountingPaID"]
resAccountingBankID	Resource HLR ID	/urf:UsageRecord/urf:Resource[@urf:description="resAccountingBankID"]
userAccountingBankID	User HLR ID	/urf:UsageRecord/urf:Resource[@urf:description="userAccountingBankID"]
resRequiresEconomicAccounting	The flag for charging	/urf:UsageRecord/urf:Resource[@urf:description="resRequiresEconomicAccounting"]
resDgasCeId	Computing Element ID	/urf:UsageRecord/urf:Resource[@urf:description="resDgasCeId"]

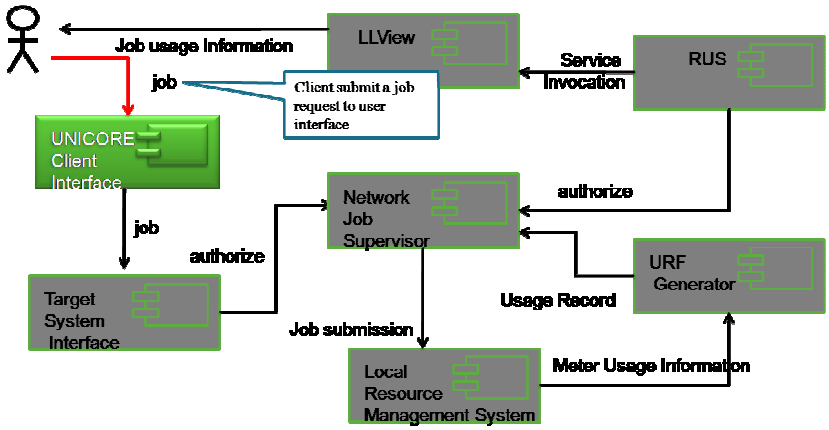
4.3 UNICORE Accounting System

4.3.1 Overview

The UNICORE Accounting System is mainly for real-time job usage monitoring. It is deployed at a single specific site only, and in terms of Usage Record has no aggregation or summarization capability. The key components are the *Usage Record Format (URF) Generator*, which meters and composes usage records, *The Resource Utilisation Service (RUS)*, and *LLView*, a usage monitoring application.

4.3.2 UNICORE Accounting System Use Case

The use case for the UNICORE Accounting System is as follows:



1. The client submits a job request through the *Client Interface*.
2. The job request is forwarded to the *Target System Interface*, which is one service of the UNICORE Atomic Services (UAS).
3. The job description (JDL) is then passed to the *Network Job Supervisor* service to be parsed and interpreted. The NJS is also in charge of authorisation.
4. If authorised, the NJS sends the parsed job request file to the *Local Resource Management System* for job submission.
5. User can monitor submitted jobs through the *LLView* monitoring application.
6. The *LLView* application invokes the *RUS* service, which is embedded within the UAS service, but at a higher level to other services.
7. The *RUS* service then delegates the request to the *Network Job Supervisor* for authorisation.
8. The *Network Job Supervisor* parses the *RUS* request into a usage record request and forwards it to the *URF Generator*.
9. The *URF Generator* meters usage information of current executing jobs.

4.3.3 UNICORE Schema and UR

[Need more here!] UNICORE is URv1 compatible, extensions?

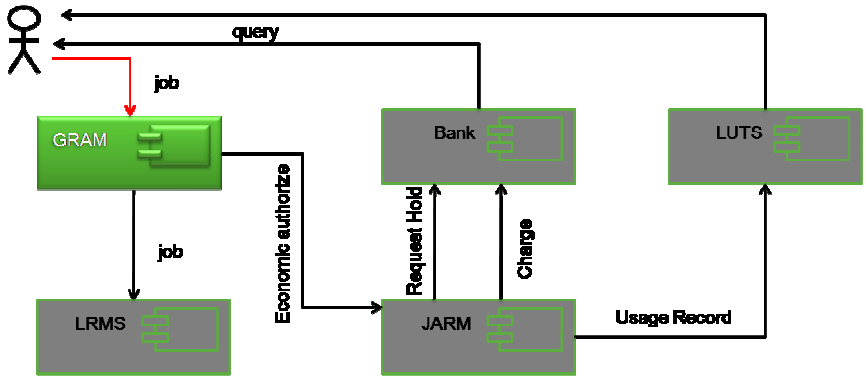
4.4 SGAS

4.4.1 Overview

SGAS (SweGrid Accounting System) is a Globus accounting component, widely deployed within Norway for economic accounting purposes. Its key components are the *Job Account Reservation Manager* for economic authorisation, the *Bank Service*, and the *Logging and Usage Tracking Service* which contains an RUS implementation for usage query.

4.4.2 SGAS Use Case

The use case for SGAS is as follows:



1. The user submits a job request to the *Globus Resource Allocation and Management (GRAM)* service.
2. Economic authorisation is then requested for the user from the *Job Account Reservation Manager (JARM)*.
3. The *JARM* requests authorisation from the *Bank*, which issues a time-limited reservation to run the job.
4. If economic authorisation is given, GRAM submits the job to the *Local Resource Management System (LRMS)*.
5. Following execution, the *JARM* then sends the economic 'charge' to the *Bank*, and submits a completed usage record to the *Logging and Usage Tracking Service (LUTS)*.
6. The user is able to query both the *Bank* and *LUTS* for economic and usage records respectively.

4.4.3 SGAS Schema and UR

SGAS additionally supports the following extension to UR:

Extension	Description	OGF UR Mapping
VOName	Virtual Organization Name	/urf:UsageRecord/urf:Resource[@urf:description="VOName"]

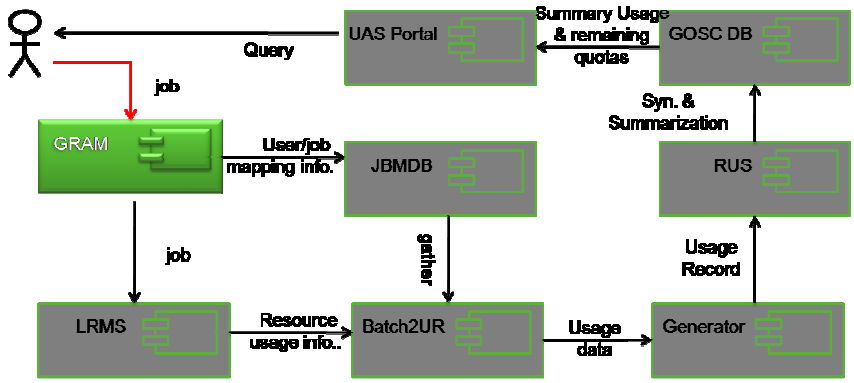
4.5 UAS

4.5.1 Overview

The UAS (User Accounting System) is used by the National Grid Service (NGS) in the UK, mainly for usage monitoring against allocated quotas on per-user basis. Its key components are the *JDMDB* module, which captures user/job mapping information by parsing Globus log files, *Batch2UR* which meters local resource usage, *UR Generator*, *RUS* and the *User Accounting System (UAS) Portal*.

4.5.2 UAS Use Case

The use case for UAS is as follows:



1. The user submits a job request to the *Globus Resource Allocation and Management (GRAM)* service.
2. *GRAM* informs the *JBMDB* about the user/job information.
3. *GRAM* submits the job to the *Local Resource Management System (LRMS)*.
4. *Batch2UR* monitors the usage of the job on the *LRMS*, and together with user/job information retrieved from *JBMDB*, generates usage data.
5. The *UR Generator* converts this usage data into usage records which are forwarded to the *Resource Utilisation Service (RUS)*.
6. The *RUS* stores usage records persistently in an Oracle relational database.
7. Usage data are summarised into the *Grid Operations Support Centre (GOSC)* database through an Oracle synchronisation mechanism.
8. Summary of use and remaining quotas can be retrieved by the *UAS Portal* on request.

4.5.3 UAS Schema and UR

The following extensions to the UR schema are supported:

Extension	Description	OGF UR Mapping
WallTimeRequested	Wall clocked requested	/urf:UsageRecord/urf:TimeDuration[@urf:description="WallTimeRequested"]
CpuTimeRequested	Cpu Duration requested	/urf:UsageRecord/urf:TimeDuration[@urf:description="CpuTimeRequested"]
TimeGlobusSubmitted	The submission time by globus gatekeeper	/urf:UsageRecord/urf:TimeInstant[@urf:description="TimeGlobusSubmitted"]
PbsLogDate	The pbs log timestamp	/urf:UsageRecord/urf:TimeInstant[@urf:description="PbsLogDate"]

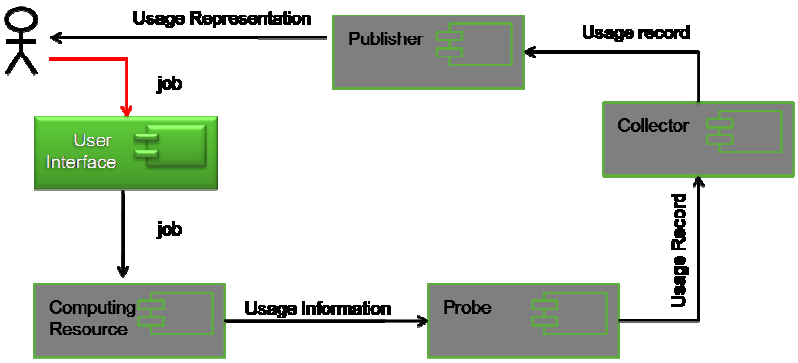
4.6 Gratia

4.6.1 Overview

Gratia is the accounting system used within the Open Science Grid (OSG), and is mainly used for usage reporting. Its key components are the *Probe*, which meters and generates usage record files, the *Collector*, which gathers usage records, and the *Publisher*, which is essentially a web portal.

4.6.2 Gratia Use Case

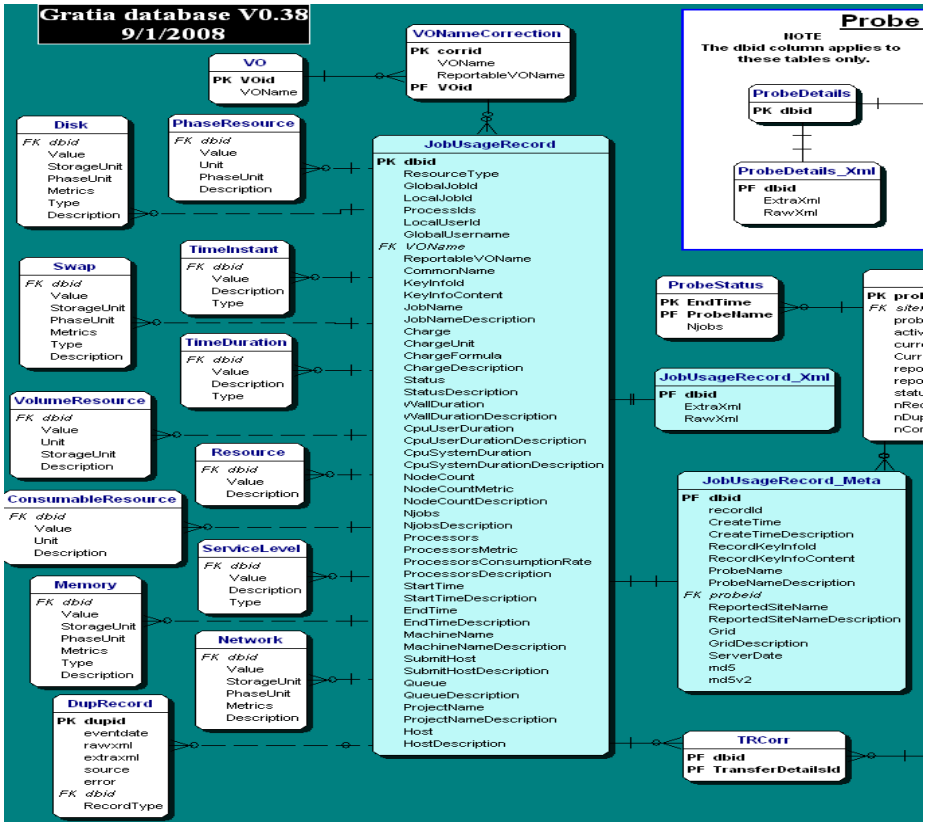
The use case for Gratia is as follows:



1. The user submits a job through the *User Interface*.
2. The job is submitted to the local *Computing Resource*.
3. Usage information is gathered by *Probe*, converted into a usage record and sent to the *Collector*.
4. The *Collector* gathers usage records and sends them on request to the *Publisher* web portal.
5. The user can query usage via the *Publisher* web portal.

4.6.3 Gratia Schema and UR

The Gratia database model is as follows:



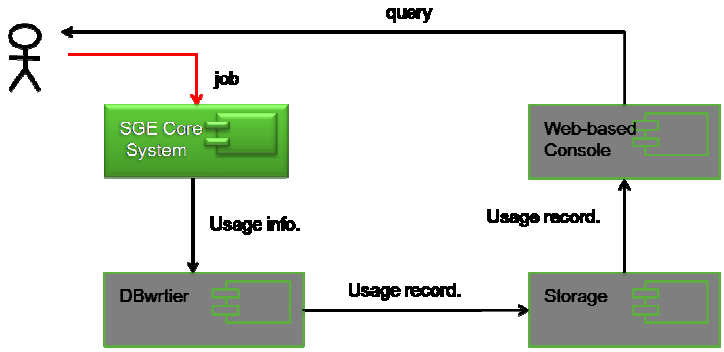
4.7 ARCO

4.7.1 Overview

ARCO (Accounting and Reporting Console) is specifically used within the Sun GridEngine, mainly for usage reporting. Its key components are the DBWriter, which meters resource usage and the *Web-based Console* tool, which is a graphical reporting portal.

4.7.2 ARCO Use Case

The ARCO use case is as follows:



1. The user submits a job to the *Sun GridEngine (SGE) Core System*.
2. The *Sun GridEngine (SGE) Core System* executes the job on the local resource.
3. The usage information is formatted into a usage record by the *DBWriter* and sent into *Storage*.
4. Upon request from the user, the *Web-based Console* can retrieve usage records from *Storage*.

4.7.3 ARCO Schema and UR

The usage schema for UR adopted by ARCO is as follows:

ARCO Schema		UR Metric	
Name	Datatype (SQL)	Context Node	Datatype (XSD)
j_id	INTEGER	/urf:UsageRecord/urf:RecordIdentity/@recordId	xsd:string
j_job_number	INTEGER	/urf:UsageRecord/urf:LocalJobId	xsd:string
j_task_number	INTEGER	/urf:UsageRecord/urf:ProcessId	xsd:string
j_owner	TEXT	/urf:UsageRecord/urf:GlobalUserName	xsd:string
j_job_name	TEXT	/urf:UsageRecord/urf:JobName	xsd:string
ju_slots	TEXT	/urf:UsageRecord/urf:Charge	xsd:double
ju_exit_status	INTEGER	/urf:UsageRecord/urf:Status	xsd:int
ju_ru_utime	DOUBLE	/urf:UsageRecord/urf:CpuDuration[@urf:usageType="user"]	xsd:duration
ju_ru_wallclock	INTEGER	/urf:UsageRecord/urf:WallDuration	xsd:duration
ju_ru_stime	DOUBLE	/urf:UsageRecord/urf:CpuDuration[@urf:usageType="system"]	xsd:duration
ju_end_time	TIMESTAMP	/urf:UsageRecord/urf:EndTime	xsd:dateTime
ju_start_time	TEXT	/urf:UsageRecord/urf:StartTime	xsd:dateTime
ju_hostname	TEXT	/urf:UsageRecord/urf:Host	xsd:string
ju_qname	TEXT	/urf:UsageRecord/urf:Queue	xsd:string
j_project	TEXT	/urf:UsageRecord/urf:ProjectName	xsd:string
ju_ru_io	DOUBLE	/urf:UsageRecord/urf:Network	xsd:int
ju_mem	DOUBLE	/urf:UsageRecord/urf:Memory[@urf:type="physical"]	xsd:int
ju_maxvmem	DOUBLE	/urf:UsageRecord/urf:Memory[@urf:type="virtual"]	xsd:int
ju_ru_nswap	INTEGER	/urf:UsageRecord/urf:Swap	xsd:int
j_submission_time	TIMESTAMP	/urf:UsageRecord/urf:TimeInstant[@description="submission_time"]	xsd:dateTime

1. Security Considerations

Please refer to RFC 3552 [RESCORLA] for guidance on writing a security considerations section. This section is required in all documents, and should not just say "there are no security considerations." Quoting from the RFC:

> "Most people speak of security as if it were a single monolithic property of a protocol or system, however, upon reflection, one realizes that it is clearly not true. Rather, security is a series of related but somewhat independent properties. Not all of these properties are required for every application.
>
> We can loosely divide security goals into those related to protecting communications (COMMUNICATION SECURITY, also known as COMSEC) and those relating to protecting systems (ADMINISTRATIVE SECURITY or SYSTEM SECURITY). Since communications are carried out by systems and access to systems is through communications channels, these goals obviously interlock, but they can also be independently provided."

2. Authors

Steve Crouch, OMII-UK, University of Southampton, s.crouch@omii.ac.uk.
Donal Fellows, University of Manchester, donal.k.fellows@manchester.ac.uk.

3. Contributors

The authors gratefully acknowledge the contributions made to this experiences document by …

4. Intellectual Property Statement

5. Disclaimer

6. Full Copyright Notice

7. References

Note that only permanent documents should be cited as references. Other items, such as Web pages or working groups, should be cited inline (i.e., see the Open Grid Forum, http://www.ogf.org). References should conform to a standard such as used by IEEE/ACM, MLA, Chicago or similar. Include an author, year, title, publisher, place of publication. For online materials, also add a URL. It is acceptable to separate out “normative references,” as IETF documents typically do. Some sample citations:

[BRADNER1] Bradner, S. Key Words for Use in RFCs to Indicate Requirement Levels, RFC 2119. March 1997.

[BRADNER2] Bradner, S. The Internet Standards Process – Revision 3, RFC 2026. October 1996.

[CATLETT] Catlett, C. GFD-1: Grid Forum Documents and Recommendations: Process and Requirements. Lemont, Illinois: Global Grid Forum. April 2002.

[RESCORLA] Rescorla, E. Guidelines for Writing RFC Text on Security Considerations. RFC 3552. July 2003.