GWD-R
UR-WG
ur-wg@ogf.org

A. Cristofori, INFN-CNAF
J. K. Nilsen, Univ. of Oslo
January 2013

Usage Record – Format Recommendation

Status of This Document

Group Working Draft (GWD)

Obsoletes

This document obsoletes GFD-R-P.098[1].

Copyright Notice

Abstract

For resources to be shared, sites must be able to exchange basic accounting and usage data in a common format. This document describes a common format with which to exchange basic accounting and usage data on different resources. This record format is intended to facilitate the sharing of usage information, particularly in the area of job accounting, storage accounting and cloud accounting but with a structure that allows an easy extension to other resources. This document describes the Usage Record components both in natural language form and annotated XML.

This document does not address how these records should be used, nor does it attempt to dictate the format in which the accounting records are stored at a local site; instead, it defines a common exchange format. Furthermore, nothing is said regarding the communications mechanisms employed to exchange the records, i.e. transport layer, framing, authentication, integrity, etc.

Contents

1 Introduction

In order for resources to be shared, sites must be able to exchange basic accounting and usage data in a common format. This document focuses on the representation of resource consumption data. The document then goes on to describe an XML-based format for usage records. The record format is intended to be specific enough to facilitate information sharing among grid sites, yet general enough that the usage data can be used for a variety of purposes: traditional usage accounting, charging, service usage monitoring, performance tuning, etc. The purpose of this document is to outline the basic building blocks of the accounting record, and how to properly represent them. All other tangential concerns such as the use, transport mechanism, and security are out of scope for this representation layer.

1.1 Context

To comprehend the structure of the schema presented in this document, it is important to understand the context in which this specification has been developed. The accounting of different use-cases involves recording:

- General properties related to the record itself
- Properties related to the consumer of the resources
- Usage of one or more resources.

Hence, the usage record schema is made up of a set of blocks for general properties, consumer and distinct resources.

1.1.1 History

Before the definition of UR-2.0, different usage record definitions building on the job accounting definition of UR-1.0[1], started to surface to describe various resource usages, such as the EMI Compute Accounting Reccord (CAR)[2], the EMI Storage Accounting Record (StAR)[3] and the EGI Cloud Usage Record (CUR)[1]. The definition of UR-2.0 came from the experience of these record definitions and is built as an easily extensible superset of these new usage records.

1.1.2 What This Document Is Not

This document and specification do not attempt to define a comprehensive "grid accounting" standard. As with all accounting implementations, there is no one-size-fits-all solution, that

[1] https://wiki.egi.eu/wiki/Fedcloud-tf:WorkGroups:Scenario4

will meet the needs of all projects and resource providers. This document does not address summary records, "grid job" records, consolidated records, or anything other than an atomic resource consumption instantiation. Sufficient resource and user information is collected to allow for effective and appropriate levels of aggregation, consolidation, and summarization, but the details of how sites implement these features (e.g., what grids do with the atomic data) are beyond the scope of this document.

1.2 Format of the Record Specification

1.2.1

2 Conventions Used in this Document

2.1 Notational Conventions

The key words "MUST" "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [4], except that the words do not appear in uppercase.

2.2 Meta Properties

Meta properties are associated with individual base properties to provide additional information and semantic meaning of the value for a base property. The meta properties outlined below are commonly encountered and should be supported for the indicated base properties.

2.2.1 Description

The description provides a mechanism for additional, optional information to be attached to a Usage Record base property. The value of this meta-property MAY provide clues to the semantic context to use while interpreting or examining the value of the owning base property.

2.2.2 Metric

This meta-property identifies the type of measurement used for quantifying the associated resource consumption if there are multiple methods to measure resource usage. As an example, disk usage may be measured as total, average, minimum or maximum usage. However, even if pertinent to the assessed charge, this meta-property does not attempt to differentiate between requested and utilized quantities of resource usage.

2.2.3 Time Stamps

Time stamps should follow the ISO 8601[5] standard as well. This includes enumerating the time zone, as specified in the standard.

2.3 Conventions

Unless otherwise stated, all fields are optional. A required field is only required if the corresponding block is present.

2.4 Supported Data Types

1. String: Data of this type has no required restrictions on the length or available characters.
2. Integer
3. Positive Integer: Data of this type must have a value of zero or greater.
4. Non-zero Integer: Data of this type must have a value of one or greater.
5. Float
6. Timestamp: Data of this type must comply with the UTC time zone format specified in ISO 8601.
7. DomainName: Data of this type must comply with RFC 1034[6] format for fully qualified domain names. The constraints are a maximum 255 characters long, containing only alphabetic and numeric characters, the "-", and the "." characters.

3 RecordIdentityBlock

This block contains the properties related to the identity of the usage record itself.

Example

```
<ur:RecordIdentityBlock>
<!| Identity properties go in here -->
</ur:RecordIdentityBlock>
```

3.1 RecordId

A record identity uniquely defines a record in the set of all usage record for the grid implementation.

- The RecordId attribute MUST be present in the RecordIdentityBlock.
- The RecordId MUST have the type string.

3.2 CreateTime

This attribute is used to specify the time when this particular Usage Record was created.

- The CreateTime attribute MUST be present in the RecordIdentityBlock.
- The CreateTime attribute MUST be an ISO timestamp.

3.3 Site

This property describes the site at which the resource is located. This property should contain a descriptive name of the group of resources which are accounted for in the record. The Site value should be constructed in such a way that it is unique within the context where it is used.

- The Site field type MUST be a string.

Example

```
<ur:Site>ACME-University</ur:Site>
```

3.4 Infrastructure

The infrastructure where the resource was used (e.g., EGI or OSG). The 'description' attribute SHOULD be used to give additional information on the used infrastructure.

- Infrastructure MUST be present

4 SubjectIdentityBlock

This block contains the properties related to the identity of the subject accounted for. Note that as all fields in this block are optional, this block SHOULD be associated with a profile that defines at least one field that must be present in the specific implementation.

Example

```
<ur:SubjectIdentityBlock>
<!| Identity properties go in here -->
</ur:SubjectIdentityBlock>
```

4.1 LocalUserId

This is the attribute property of the Usage Record is part of the IdentityBlock and identify the local user (eg.: Unix user).

- The LocalUser field type MUST be a string.

Example

```
<ur:LocalUser>johndoe</ur:LocalUser>
```

4.2 LocalGroupId

This is the attribute property of the Usage Record is part of the IdentityBlock and identify the local group (eg.: Unix group).

- The LocalGroup field type MUST be a string.

Example

```
<ur:LocalGroup>binarydataproject</ur:LocalGroup>
```

4.3 GlobalUserId

This property describes the global identity of the user accountable for the resource consumption. The property should identify the user globally, such that clashes do not happen accidentally, e.g. it could be an X500 identity.

- The UserIdentity field type MUST be a string.

4.4 GlobalGroupId

This property describes the global group accountable for the resource consumption. The property should identify the group globally, such that clashes do not happen accidentally, e.g. using a FQDN to construct it. In Grid terms, this would typically be the VO name.

- The Group field type MUST be a string.

4.5 GlobalGroupAttribute

This property describes supplemental traits of the group property, e.g., sub-groups, role or authority. This makes it possible to account for segments of a group, while still being able to account for the group as a whole. The property consists of a type which denotes the type of attribute and an actual value for the attribute.

- The GlobalGroupAttribute property can be repeated.
- The GlobalGroupId property MUST exist in the record if GlobalGroupAttribute is specified.
- The GlobalGroupAttribute type and field values MUST exist.
- The GlobalGroupAttribute type MUST have the type string.
- The GlobalGroupAttribute field type MUST be a string.

5 ComputeUsageBlock

This block contains the properties related to compute usage.

Example

```
<ur:ComputeUsageBlock>
<!|Compute Record properties go in here -->
</ur:ComputeUsageBlock>
```

5.1 CpuDuration

This element contains the CPU time consumed. If the task ran on many cores/processors/nodes, all separate consumptions shall be aggregated in this value. This has an impact for example on on MPI usage, where the consumption of all the "nodes" get aggregated into this CPU consumption.

- CpuDuration MUST be present and MUST contain a time duration as defined in ISO 8601:2004[5].

5.2 WallDuration

WallClock time elapsed during the job execution. Basically it EndTime-StartTime no matter on how many cores, processors, nodes the user task ran on. In MPI WallClock duration might be lower than CpuDuration.

- WallDuration MUST be present and MUST contain a time duration as defined in ISO 8601:2004[5].

5.3 StartTime

This property describes a timestamp indicating the time at which the measured resource consumption started. Together with EndTime this defines a period over which the resource has been consumed.

- The StartTime field MUST be present in the block.
- The StartTime field type MUST be an ISO timestamp.
- The time zone may be specified as Z (UTC) or (+—-)hh:mm. Time zones that are not specified are considered undetermined.

5.4 EndTime

This property describes a timestamp indicating the time at which the measured resource consumption ended. Together with StartTime this defines a period over which the resource has been consumed.

- The EndTime field MUST be present in the block.
- The EndTime field type MUST be an ISO timestamp.
- The time zone may be specified as Z (UTC) or (+—-)hh:mm. Time zones that are not specified are considered undetermined.

5.5 Host

This is used to identify the host where the user consumed the resource. As an example, in case of MPI jobs more than one Host parameter can be specified. If a master node for the computation can be identified, it can be flagged as "primary" with a dedicated attribute.

- Host SHOULD be present. Multiple instances of this element MAY be present.

5.6 HostType

The type of service according to a namespace-based classification. The namespace MAY be related to a middleware name, and organization or other concepts; org.ogf.glue.* is reserved for Types defined by the OGF GLUE Working Group.

5.7 Processors

The number of processors used or requested. A processor definition may be dependent on the machine architecture. Typically processor is equivalent to the number of physical CPUs used. For example, if a job uses two cluster "nodes", each node having 16 CPUs each, the total number of processors would be 32.

- The value MUST be of type Non-zero Integer

5.8 NodeCount

Number of nodes used. A node definition may be dependent on the architecture, but typically a node is a physical machine. For example a cluster of 16 physical machines with each machine having one processor each is a 16 "node" machine, each with one "processor". A 16 processor SMP machine however, is 1 physical node (machine) with 16 processors.

- The value MUST be of type Non-zero Integer

5.9 ProcessId

The UNIX pid of the job on the executing node. If all processes associated with a job are tracked as part of usage, each individual process identifier MAY be reported with this element.

- ProcessId MAY be present in the record.
- ProcessId MAY be present multiple times in the record.
- The Host attribute SHOULD be used to identify the host to which the process id refers to. For example this is required in case of MPI jobs that use multiple hosts.

Example

```
<urf:ProcessId urf:Host="compute-0-1.abel.uio.no">1042</urf:ProcessId>
```

6 JobUsageBlock

This block contains the properties related to job usage. The block is intended to be used for jobs submitted to a batch system.

Example

```
<ur:JobUsageBlock>
<!|Job Record properties go in here -->
</ur:JobUsageBlock>
```

6.1 GlobalJobId

This property describes the global identity of the job. The property should identify the job globally, such that clashes do not happen accidentally. This could for example be a combination of time stamp, local job id and host name.

- The GlobalJobId field type MUST be a string.

6.2 LocalJobId

This property describes the local identity of the job.

- This element MUST contain the LRMS ID for the job as assigned by the batch system.

6.3 JobName

This attribute may contain a descriptive of the job. It has to be stressed that user defined job names are often difficult to retrieve from an accounting perspective and are not suitable for reliable accounting purposes.

6.4 Charge

This is property may be used for economic accounting purpose. It is the charge applied to the user resource usage.

- When specifying a charge, the following two attributes MUST be specified:
  - unit – defines the currency used to report the charge.
  - formula – The cost computation formula adopted to compute the applied charge.

6.5 MachineName

Identifier for the computing facility managing the job.

- MachineName SHOULD be the LRMS server host name.
- MachineName SHOULD be present in the record.

6.6 SubmitHost

The host submitting the job.

- In a grid environment SubmitHost MUST be present
- In a grid environment SubmitHost MUST report the Computing Element Unique ID.

6.7 Middleware

The purpose of this element is to mark whether the job was submitted locally or through a grid middleware. At least the values "local" and "grid" MUST be supported. The description attribute SHOULD be used to give additional information on the used middleware.

- Middleware MUST be present.
- The values "local" and "grid" MUST be supported.
- The description attribute SHOULD be used.

6.8 Queue

The name of the queue from which the job was executed or submitted.

- Queue MUST have data of type string.
- Description MAY be specified.

6.9 TimeInstant

This element may be used to insert in the records any time instant related to the user payload. Three optional values for the type attribute are defined as they are of common usage by batch systems. The semantic is derived from Torque.

- TimeInstant SHOULD be present in the record.
- TimeInstant MAY be present multiple times.

- The following three types are defined and SHOULD be reported:
  - Ctime - Time job was created
  - Qtime - Time job was queued
  - Etime - Time job became eligible to run

6.10 ServiceLevel

This element is used to insert computing benchmarks and normalization factors. At least one normalization factor MUST be present in the record.

- ServiceLevel MUST be present in the record.
- ServiceLevel MAY be present multiple times.
- At least the following types should be supported:
  - Si2k – SpecInt2000
  - Sf2k – SpecFloat2000
  - HEPSPEC – HEPSpec

6.11 Status

This property will represent the completion status of the job. For example, this may represent the exit status of an interactive running process or the exit status from the batch queuing system's accounting record. The semantic meaning of status is site dependent.

- The value of Status MUST be of type String.
- The Status property MUST exist in the record.
- The Status property MUST support the following values:
  - aborted – A policy or human intervention caused the job to cease execution.
  - completed – The execution completed.
  - failed – Execution halted without external intervention.
  - held – Execution is held at the time this usage record was generated.
  - queued – Execution was queued at the time this usage record was generated.
  - started – Execution started at the time this usage record was generated.

  - suspended – Execution was suspended at the time this usage record was generated.

- The Status property MAY support other values, as agreed upon within the implementation context.

6.12 ExitStatus

This element allows to specify the numeric exit status value for the application.

- The ExitStatus propery MUST be present in the record.
- The value of ExitStatus MUST be of type Integer

7 MemoryUsageBlock

This block contains the properties related to memory usage. The block may be present several times to account for different types of memory (e.g., RAM and swap).

Example

Example

```
<ur:MemoryUsageBlock>
<!| Memory Usage properties go in here -->
</ur:MemoryUsageBlock>
```

7.1 MemoryClass

This property describes the class of memory used (virtual and real memory must be supported but others might be specified). This is a descriptive value, which allows the memory system to provide details about the memory used.

Example

Example

```
<ur:MemoryClass>RAM</ur:MemoryClass>
```

7.2 MemoryResourceCapacityUsed

This property describes the number of physical bytes used on the memory system (e.g., the amount of memory resources used for this process). This is the main metric for measuring memory consumption.

Example

Example

```
<ur:MemoryResourceCapacityUsed>14728</ur:MemoryResourceCapacityUsed>
```

7.3 MemoryLogicalCapacityUsed

This property describes the number of bytes allocated for this process on the memory system (e.g., the amount of memory resources made available for this process; this value may be

higher than the MemoryResourceCapacityUsed because it may also include bytes that are not really used by the process).

Example

Example

```
<ur:MemoryLogicalCapacityUsed>56437</ur:MemoryLogicalCapacityUsed>
```

7.4 MemoryResourceCapacityAllocated

This property describes the number of bytes required by the process (e.g., the memory requested in a job description).

Example

Example

```
<ur:MemoryResourceCapacityAllocated>42000</ur:MemoryResourceCapacityUsed>
```

7.5 Host

This is used to identify the host where the user consumed the resource.

- Host SHOULD be present.

7.6 StartTime

Same as StartTime in ComputeUsageBlock.

7.7 EndTime

Same as EndTime in ComputeUsageBlock.

8 StorageUsageBlock

This block contains the properties related to storage usage.

Example

Example

```
<ur:StorageUsageBlock>
<!|Storage Record properties go in here -->
</ur:StorageUsageBlock>
```

8.1 Host

This property describes the storage system on which the resources have been consumed. This value should be chosen in such a way that it globally identifies the storage system, on which resources are being consumed (eg., the FQDN of the storage system could be used).

- The Host property MUST be present in the record.
- The Host field MUST have the type string.
- The Host value SHOULD be constructed in such a way, that it globally identifies the storage system.

Example

Example

```
<ur:Host>host.example.org</ur:Host>
```

8.2 StorageShare

This property describes the part of the storage system which is accounted for in the record. For a storage system, which is split into several logical parts, this can be used to account for consumption on each of these parts. The value should be able to identity the share of the storage system, given the storage system property.

- The StorageShare field type MUST be a string.

Example

Example

```
<ur:StorageShare>pool-003</ur:StorageShare>
```

8.3 StorageMedia

This property describes the media type of storage that is accounted for in the record (eg.: "disk" or "tape". This allows for accounting of different backend storage types).

- The StorageMedia field type MUST be a string.

Example

Example

```
<ur:StorageMedia>disk</ur:StorageMedia>
```

8.4 StorageClass

This property describes the class of the stored data, e.g. "pinned", "replicated", "precious". This is a descriptive value, which allows the storage system to provide details about the stored data.

- The StorageClass field type MUST be a string.

Example

Example

```
<ur:StorageClass>replicated</ur:StorageClass>
```

8.5 DirectoryPath

This property describes the directory path being accounted for. If the property is included in the record, the record should account for all usage in the directory and only that directory.

- The DirectoryPath field type MUST be a string.

Example

Example

```
<ur:DirectoryPath>/projectA</ur:DirectoryPath>
```

8.6 FileCount

This property describes the number of files which are accounted for in the record.

- The FileCount field type MUST be a positive non-zero integer.

Example

Example

```
<ur:FileCount>42</ur:FileCount>
```

8.7 StorageResourceCapacityUsed

This property describes the number of bytes used on the storage system or storage share where appropriate. This is the main metric for measuring resource consumption. It should include all resources for which the identity of the record is accountable for.

StorageResourceCapacityUsed can include reserved space, file metadata, space used for redundancy in RAID setups, tape holes, or similar. The decision about including such "additional" space is left to the resource owner but should be made known to the user e.g. via the usage policy. In contrary the StorageLogicalCapacityUsed denotes the pure file size. If available, reserved space can be recorded explicitly with StorageResourceCapacityAllocated.

- The StorageResourceCapacityUsed property MUST be present in the record.
- The StorageResourceCapacityUsed field type MUST be a nonnegative integer.
- StorageResourceCapacityUsed SHOULD include all resources that are used to store the files. The value MAY also include resources that are no longer in use but are unavailable for reuse (e.g., if a file is removed from tape, the tape may not be immediately available for reuse), as documented in the appropriate SLA or usage policy documents.

Example

Example

```
<ur:StorageResourceCapacityUsed>14728</ur:StorageResourceCapacityUsed>
```

Implementation Note:

Using bytes saves us from the argument of discussing if 1000 or 1024 should be used as a base. However, this also means that the number reported can be very large. Therefore any

implementation should use at least a 64-bit integer to hold this variable (a signed 64-bit integer will overflow at 8 Exabytes).

8.8 StorageLogicalCapacityUsed

This property describes an integer denoting the number of "logical" bytes used on the storage system by the identity of the record. By "logical" is meant the sum of bytes of the files stored, i.e. excluding reservation and any underlying replicas of files.

- The StorageLogicalCapacityUsed field type MUST be a nonnegative integer.

Example

Example

```
<ur:StorageLogicalCapacityUsed>13617</ur:StorageLogicalCapacityUsed>
```

Implementation Note:

Same as for StorageResourceCapacityUsed property.

8.9 StorageResourceCapacityAllocated

This property describes the number of bytes allocated on the storage system or storage share where appropriate. Depending on implementation this property may be equal to StorageResourceCapacityUsed, however this property should only take into account space allocated to the entity described in the record, not resources used for redundancy in RAID setups, tape holes, or similar.

- The StorageLogicalCapacityUsed field type MUST be a non-negative integer.

Example

Example

```
<ur:StorageResourceCapacityAllocated>14624</sr:StorageResourceCapacityAllocated>
```

Implementation Note:

Same as for StorageResourceCapacityUsed property.

8.10 StartTime

Same as StartTime in ComputeUsageBlock.

8.11 EndTime

Same as EndTime in ComputeUsageBlock.

8.12 HostType

Same as HostType in ComputeUsageBlock.

9 CloudUsageBlock

This block contains the properties related to cloud usage.

Example

Example

```
<ur:CloudUsageBlock>
<!|Storage Record properties go in here -->
</ur:CloudUsageBlock>
```

9.1 VirtualMachineId

A descriptive name that (uniquely?) identifies the instance of Virtual Machine instantiated.

Example

Example

9.2 Status

- This property SHOULD be referred to as status.
- This property MUST contain data of type string.
- This property MUST exist.
- This property MUST support the following values:
  - completed – The execution completed.
  - started – Execution started at the time this usage record was generated.
  - suspended – Execution was suspended at the time this usage record was generated.
- This property MAY support other values, as agreed upon within the implementation context.

Example

Example

9.3 SuspendTime

SuspendTime is a value that count fot the time that record the moment in which the Virtual Machine changed status to "suspended".

- SuspendTime MUST be present when Status is set to "suspended".
- SuspendTime field type MUST be an ISO timestamp.
- time zone may be specified as Z (UTC) or (+—-)hh:mm. Time zones that are not specified are considered undetermined.

9.4 ImageId

ImageId identify provides the ID of the image used to instanciate the VM associated with the record.

9.5 HostType

Same as HostType in ComputeUsageBlock.

9.6 TimeInstant

Same as TimeInstant in JobUsageBlock.

10 Attributes Matrix

UR 2.0	**UR 1.0**	**StAR 1.0**	**CAR 1.0**	**Cloud Accounting Usage Record**	**GLUE 2.0**
RecordIdentityBlock		**RecordIdentity**	**RecordIdentity**		
recordId	RecordIdentity	recordId	recordId	RecordId/ StorageRecordId	
createTime		createTime			
Infrastructure			Infrastructure		
Site		Site	Site	Site	AdminDomain.Name
				ZoneName	
				TimeZone	
SubjectIdentityBlock		**SubjectIdentity**	**UserIdentity**		
LocalUserId	LocalUserId	LocalUser	LocalUserId	LocalUserId	
LocalGroupId		LocalGroup	LocalGroup	LocalGroupId	
GlobalUserId		UserIdentity	GlobalUserName	GlobalUserName	
GlobalGroupId	GlobalUsername	Group	Group	FQAN	
GlobalGroupAttribute	ProjectName	GroupAttribute	GroupAttribute		
ComputeUsageBlock			**JobIdentity**		
CpuDuration	CpuDuration		CpuDuration	CpuDuration	
WallDuration	WallDuration		WallDuration	WallDuration	
StartTime	StartTime		StartTime	StartTime	
EndTime	EndTime		EndTime	EndTime	
Host		Host			
NodeCount	NodeCount		NodeCount		
Processors	Processors		Processors	CpuCount	
ProcessId	ProcessId		ProcessId		
HostType					ServiceType_t
JobUsageBlock			**JobIdentity**		
GlobalJobId	GlobalJobId		GlobalJobId		
LocalJobId	LocalJobId		LocalJobId		
JobName	JobName		JobName		
Charge	Charge		Charge		
Status	Status		Status		
ExitStatus			ExitStatus		
MachineName	MachineName		MachineName		
Middleware					

SubmitHost	SubmitHost		SubmitHost		
Queue	Queue		Queue		
	TimeDuration		TimeDuration		
TimeInstant	TimeInstant		TimeInstant		
ServiceLevel	ServiceLevel		ServiceLevel		
	Extensions		Extensions		
MemoryClass	Memory/Swap		Memory/Swap	Memory	
Memory-Resource-CapacityUsed					
MemoryLogical-CapacityUsed	Memory/Swap		Memory/Swap	Memory	
Memory-Resource-Capacity-Allocated					
Host					
StartTime					
EndTime					
HostType					ServiceType_t
StorageUsage-Block					
Host		StorageSystem			
StorageShare		StorageShare			
StorageMedia	(Disk)	StorageMedia			
StorageClass		StorageClass			
DirectoryPath		DirectoryPath			
FileCount		FileCount			
Storage-Resource-CapacityUsed	Disk	Resource-CapacityUsed		Disk	
StorageLogical-CapacityUsed	Disk	LogicalCapacity-Used			
Storage-Resource-Capacity-Allocated		Resource-Capacity-Allocated			
StartTime		StartTime			
EndTime		EndTime			
HostType					ServiceType_t
CloudUsage-Block					
VirtualMachineId				MachineName	
Status				Status	
				SuspendTime	

SuspendTime					
ImageId				ImageId	
TimeInstant					
HostType				CloudType	ServiceType_t
Network-UsageBlock					
				NetworkType	
			Network	NetworkInbound	
			Network	Network-Outbound	

11 Field Summaries

12 Examples

13 Security Considerations

Please refer to RFC 3552 [**?**] 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."

14 Glossary

Recommended but not required.

15 Contributors

Contact information for authors. You can also use this section to recognize contributions by other people who are not listed as authors, but made a useful contribution.

The title page should list the Corresponding Authors (or Editors), who are committed to taking permanent stewardship for this document – receiving communication in the future and otherwise being responsive to its content. Corresponding authors will be sought to process any error reports. The title page should contain at least one and at most three (Corresponding) Author/Editors, unless there are compelling reasons to list more.

Corresponding authors must be indicated as part of the Contributors or Authors section. Contributors are individuals who assisted with a document's preparation, and whose contributions are recognized in the document.

The OGF prefers the use of full first names (not initials). Complete contact information for authors must be included. Contributors are listed after authors, and do not need to have complete contact information. The nature of the contribution may be recognized.

Andrea Cristofori
(Corresponding author)
Institution1
Address
Country
Email: jdoe@example.com

Jon Kerr Nilsen
(Corresponding author)
Institution1
Address
Country
Email: jdoe@example.com

John Gordon
Institution1
Address
Country
Email: jdoe@example.com

Alison Paker
Institution1
Address
Country
Email: jdoe@example.com

John Alan Kennedy
Institution1
Address
Country
Email: jdoe@example.com

Ralph Müller-Pfefferkorn
Institution1
Address
Country
Email: jdoe@example.com

16 Acknowledgments

Include if desired. Contributors to the document may also be listed in the previous section.

Authors of CAR, StAR, CUR. People who commented in StAR public hearing.

17 Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

18 Disclaimer

This document and the information contained herein is provided on an "As Is" basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

19 Full Copyright Notice

as needed for the purpose of developing new or updated OGF documents in conformance with the procedures defined in the OGF Document Process, or as required to translate it into languages other than English. OGF, with the approval of its board, may remove this restriction for inclusion of OGF document content for the purpose of producing standards in cooperation with other international standards bodies.

The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or assignees.

20 References

[1] RR Mach, R. Lepro-Metz, S. Jackson, and L. McGinnis. Usage record format recommendation gfd-rp. 098. In *Open Grid Forum Recommendation*, 2007.

[2] A. Guarise. Definition of the compute accounting record (car), 2011. URL `http://cdsweb.cern.ch/record/1449764`.

[3] JK Nilsen, P. Millar, R. Müller-Pfefferkorn, Z. Molnar, and R. Zappi. Emi star–definition of a storage accounting record. 2011.

[4] Scott Bradner. Key words for use in RFCs to Indicate Requirement Levels. RFC 2119 (Best Current Practice), March 1997. URL `http://tools.ietf.org/html/rfc2119`.

[5] M. Wolf and C. Wicksteed. Date and time formats. *W3C NOTE NOTE-datetime-19980827, August*, 1998.

[6] P. Mockapetris. Rfc 1034: Domain names-concepts and facilities, 1987. *URL: ftp://ftp. isi. edu/in-notes/rfc1034. txt.*