draft-ggf-cgs-jsim_spec-8sep2003.doc
CIM Grid Schema Workgroup
http://forge.gridforum.org/projects/cgs-wg

Editors:
E. Stokes, IBM
L. Flon, USC/ISI

8 September 2003

Job Submission Information Model (JSIM)

Version 1.0

Status of This Memo

This document provides information to the community regarding the specification of the Job Submission Information Model (JSIM), based on the DMTF's Common Information Model. Distribution of this document is unlimited.

Abstract

This document describes the Job Submission Interface Model. It is based on the "job" schema in DMTF's Common Information Model (CIM), version 2.8 preliminary [CIM2.8prelim]. It includes a UML diagram of the classes associated with job submission, the managed object format (MOF) for those classes.

Editors:
E. Stokes, IBM
L. Flon, USC/ISI

8 September 2003

GLOBAL GRID FORUM

office@gridforum.org
www.ggf.org

Full Copyright Notice

Intellectual Property Statement

The GGF 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 GGF Secretariat.

The GGF 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 GGF Executive Director (see contact information at GGF Web site).

Editors:
E. Stokes, IBM
L. Flon, USC/ISI

8 September 2003

Contents

1. Overview

The Job Submission Information Model (JSIM) describes the managed objects and their relationships for managing the execution and monitoring of batch jobs in a grid environment. The CIMv2.8 preliminary schema for jobs and processing is the foundation for the development of this model.

It is our recommendation that grid (batch) scheduling systems act as information providers to a CIMOM (CIM Object Manager), so that any CIM browser can be used to inspect the status of queues and jobs on a grid resource, regardless of the brand of scheduler being utilized. The model defined herein represents our current thinking on the CIM extensions necessary to represent this data.

In a related document we will also suggest, as a strawman, an XML schema that could be used by OGSI-compliant Grid Services to exchange the information in the model presented.

The keywords “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED” and “MAY” used in this document are to be interpreted as described in [RFC2119].

2. Introduction to the Model

The new elements of the model (on top of base CIM 2.8 preliminary schema classes) are: BatchService, BatchSAP, BatchJob, BatchJobGroup, QueueForBatchService, ExecutionCandidate, ExecutionTarget, ExecutionRequirement, and OwningCollectionElement.[1,2]

The classes and attributes added to CIM 2.8 preliminary in support of the JSIM work are marked as ‘Experimental’ in the CIM MOF and as {E} in the UML diagram.

The basic premise of JSIM is that a BatchService (there can be one more more hosted in a single CIMOM) represents the root of information for a job scheduling system. A BatchService can host any number of JobQueues, via the association QueueForBatchService.

The jobs on a JobQueue are represented by a Batch Job or a BatchJobGroup, which is a CIM Collection. Each item of the BatchJobGroup is a BatchJob. The BatchJob class inherits from CIM ConcreteJob and CIM Job, and we have added associations ExecutionCandidate and ExecutionTarget between Job and System to show the relationship between a batch job and one or more nodes capable of running the job and actually running the job on one or more nodes, respectively.

It is thus possible to locate all of the BatchService’s in the CIMOM, all of their JobQueues, and the BatchJobs on those queues. It is likewise possible, from the point of view of a BatchJob, to find the JobQueue and BatchService it is associated with.

Related aspects of the model include ConcreteCollection, which is a CIM Collection of CIM Systems. This facilitates the allocation of a job scheduler to a group of computer systems in a cluster, a group of processors on a single computer, etc.

[1] These elements will be submitted to DMTF for inclusion in CIM V2.9.
[2] We intend to propose that the associations ExecutionCandidate, ExecutionTarget, ExecutionRequirement, and OwningCollectionElement should really be part of CIM V2.8, that is, are of general use to job processing. We have the opportunity to propose modifications to CIM V2.8 prelim for CIM V2.8 final until early November 2003.

Both Job and JobSettingData can include recovery action data.

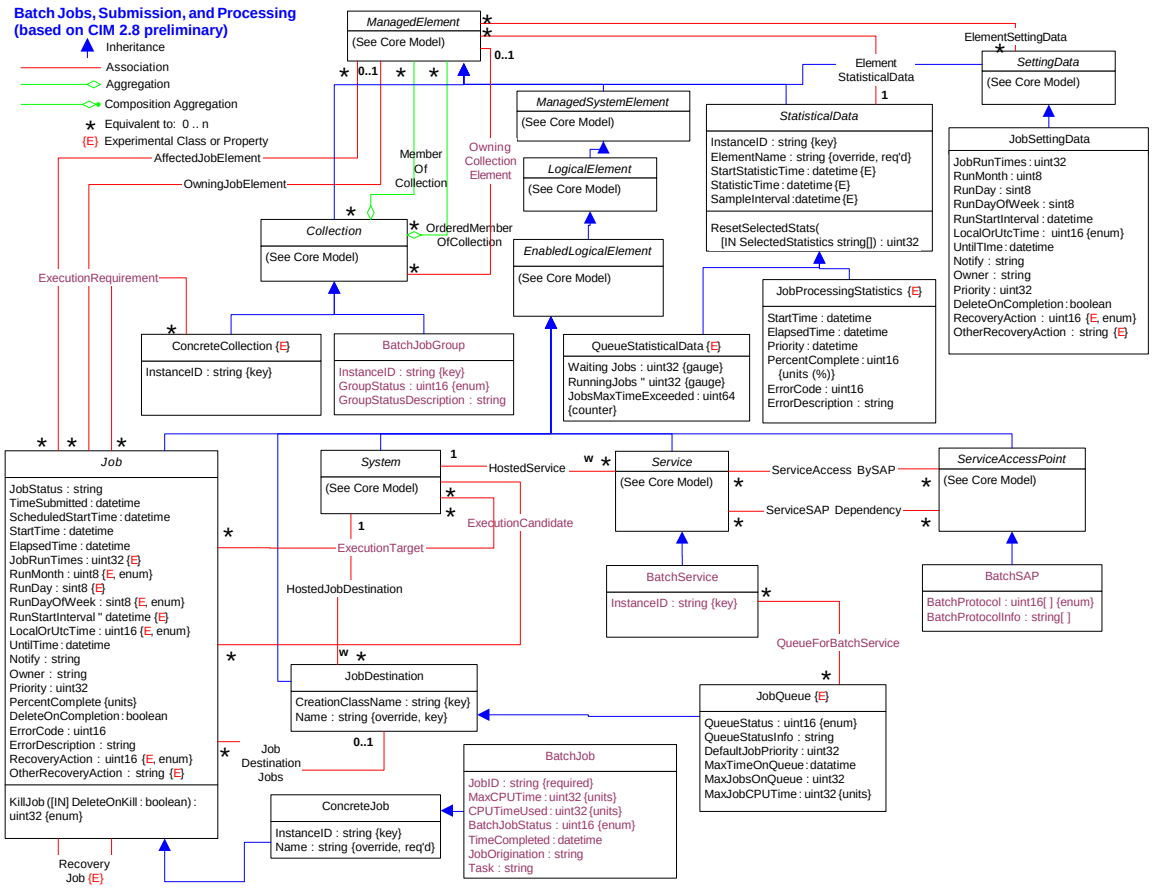
BatchSAP specifies the batch submission procotols that BatchService uses. The relationship is through the association ServiceAccessBySAP.

The model elements are discussed in a bit more detail in Section 4 following the presentation of the UML in Section 3. The ManagedObject Format (MOF) description is given in Section 5.

This JSIM does not address nor specifically depend upon any resource model, e.g. cluster computing model. Nor does it address or depend upon any particular software or method for submitting jobs for execution. It contains the standard information and relationships that allow a wide range of grid execution systems to operate efficiently and effectively.

3. UML

The figure below depicts the classes and properties of the batch job schema. The new classes and properties are highlighted in this color (plum).



4. Discussion of the Model Elements

This section lists the classes, associations, and properties beyond that provided in CIM 2.8 preliminary.

4.1. Classes Added

4.1.1. BatchJob

The BatchJob class inherits from the CIM_ConcreteJob (and thence Job), with the specific meaning that, while ConcreteJob refers to a job executing on the System, a BatchJob is on a batch queue and traceable in that fashion.

Contents:

- JobID (required)
  An arbitrary string (hopefully uniquely) identifying this job.

- MaxCPUTime
  The maximum amount of execution time on CPU this job is allowed.

- CPUTimeUsed
  The amount of execution time on CPU currently consumed by this job.

- BatchJobStatus
  Describes the current state of this BatchJob with respect to the BatchQueue and the system executing this job.

- TimeCompleted
  When this job finished.

- JobOrigination
  Provides additional information, beyond Job Owner inherited from CIM_Job, to identify the origins of the BatchJob. This property could include information such as the System, application or Process that created the BatchJob.

- Task
  The execution command for this job.

4.1.2. BatchJobGroup

A BatchJobGroup inherits from CIM_Collection. BatchJobs are related through the associations OwningCollectionElement and OrderedMemberOfCollection.

Contents:

- InstanceID
  An opaque string uniquely identifying this job.

- GroupStatus:
  The current status of this BatchJobGroup, based on the statuses of the jobs contained in the Group. The values of GroupStatus are more general than those of the individual jobs

- GroupStatusDescription
  GroupStatusDescription provides additional information regarding the GroupStatus property.

4.1.3. BatchService

A BatchService inherits from CIM_Service, and especially refers to a job scheduler system.

Contents:

- InstanceID
  An opaque string uniquely identifying this service.

4.1.4. BatchSAP

This class inherits from CIM_ServiceAccessPoint, and specifies the protocols used to interact with the job scheduler service. The protocols are generic in this specification.
Contents:

- BatchProtocol
  A number connoting the specified protocol.

- BatchProtocolInfo
  An arbitrary string containing information specific to this protocol.

4.2. New Associations

4.3.1. ExecutionTarget
This identifies the System that is running the Job.

4.3.2. ExecutionCandidate
This identifies the System that is capable of running the Job.

4.3.3. QueueForBatchService
This ties one or more JobQueues to a BatchService.

4.3.4. OwningCollectionElement
This identifies the ManagedElement owning the Collection. In JSIM, the BatchService is owning/controlling the one or more ConcreteCollection.

4.3.5. ExecutionRequirement
This identifies the Job that was submitted to one or more ConcreteCollection.

5. Managed Object Format (MOF)

The schema is described in Managed Object Format, defined in [CIMspec].

The MOF below reflects the UML diagram in this document. There may be errors in it, but it is included to provide the details and descriptions necessary to understand the UML.

```
// ===================================================================
// Title:       batch job, based on CIM 2.8 preliminary
// Filename:    batch job.mof
// Version:
// Release:
// Date:        27 August 2003
```

```
// ===================================================================
// Description: This file defines the classes to manage batch job
//              submission.  It builds on the CIM 2.8 preliminary
//              schema for jobs and processing.
//              *** targeted for CIM 2.9 preliminary ***
// ===================================================================
// Change Log
// Feb 25 2003 - Initial release
// Feb 28 2003 - Added Key property to BatchJobGroup
//     Clarified Descriptions for BatchJob and BatchJob.BatchJobStatus
// Jun 6 2003  - Lots of changes
// Jun 11 2003 - Added ExecutionRequirement association
// Jun 26 2003 - changes from GGF CGS WG session
// Aug 27 2003 - Modified to supplement CIM 2.8 prelim
//
// ===================================================================

#pragma locale ("en_US")


// ===================================================================
// BatchJob
// ===================================================================
[Experimental, Version ("2.9.0"), Description (
    "Description of a batch request that is either waiting on a "
    "BatchQueue to run, in the process of running, or that has "
    "previously run. Jobs that are completed and not recurring "
    "will NOT be associated with a BatchQueue.  Jobs that are "
    "completed but recurring WILL be associated with a BatchQueue, "
    "since they are waiting to run given the scheduling "
    "information in the associated JobScheduleSettingData instance.") ]
class CIM_BatchJob : CIM_ConcreteJob {

    [Required, Description (
        "Uniquely identifies this Job within a scoping BatchQueue, "
        "and BatchService. This property can be used in the "
        "construction of the InstanceID key.") ]
    string JobID;

    [Description (
        "Specifies the maximum number of milliseconds of CPU "
        "time that this job can use."),
        Units("MilliSeconds") ]
    uint32 MaxCPUTime;

    [Description (
        "Specifies the number of milliseconds of CPU that this "
        "job has used.  This number will continue to change until "
        "the job has finished its execution, either successfully "
        "or unsuccessfully."),
        Units("MilliSeconds") ]
    uint32 CPUTimeUsed;

    [Description (
        "Describes the current state of this BatchJob with respect "
        "to the BatchQueue and the system executing this job. "
        "Additional information may be specified in JobStatus."),
```

```
        ValueMap {"0", "1", "2", "3", "4", "5", "6"},
        Values {"Unknown", "Other", "Pending", "Blocked",
            "Complete", "Completed With Error", "Running"},
        ModelCorrespondence {"CIM_BatchJob.TimeCompleted",
            "CIM_BatchJob.JobStatus"} ]
    uint16 BatchJobStatus;

    [Description (
        "Time when this BatchJob was completed. This value is only "
        "valid if the BatchJobStatus has been set to \"Complete\" "
        "(value=5) or \"Completed With Error\" (value=6)."),
        ModelCorrespondence {"CIM_BatchJob.BatchJobStatus"} ]
    datetime TimeCompleted;

    [Description (
        "Provides additional information, beyond Job Owner "
        "inherited from CIM_Job, to identify the origins of the "
        "BatchJob. This property could include information such as "
        "the System, application or Process that created the "
        "BatchJob.") ]
    string JobOrigination;

    [Description (
      "Provides the command and parameters, in string form, for "
      "the execution of this job.") ]
    string Task;
};


// ==================================================================
// BatchJobGroup
// ==================================================================
[Experimental, Version ("2.9.0"), Description (
    "BatchJobGroup describes a collection of BatchJobs that are "
    "logically grouped.  Two grouping/scheduling concepts are "
    "modeled: (1) User grouping: Jobs are logically grouped by "
    "the user of the batch system to capture common "
    "characteristics.  There is no sequencing relationship "
    "between the jobs belonging to the Group. The jobs are grouped "
    "using the MemberOfCollection association. (2) Job "
    "sequencing: An ordered collection of jobs is defined "
    "where the jobs are executed in sequence. This captures a simple "
    "sequencing relationship for a set of jobs and is defined "
    "using the OrderedMemberOfCollection association. An individual "
    "BacthJob can belong to multiple BatchJobGroups.") ]
class CIM_BatchJobGroup : CIM_Collection {

    [Key, Description (
        "Within the scope of the instantiating Namespace, InstanceID "
        "opaquely and uniquely identifies an instance of this class. "
        "In order to ensure uniqueness within the NameSpace, the "
        "value of InstanceID SHOULD be constructed using the "
        "following 'preferred' algorithm: \n"
        "   <OrgID>:<LocalID> \n"
        "Where <OrgID> and <LocalID> are separated by a colon ':', "
        "and where <OrgID> MUST include a copyrighted, trademarked "
```

```
        "or otherwise unique name that is owned by the business entity
"
        "creating/defining the InstanceID, or is a registered ID that "
        "is assigned to the business entity by a recognized global "
        "authority. (This is similar to the <Schema Name>_<Class Name>
"
        "structure of Schema class names.) In addition, to ensure "
        "uniqueness <OrgID> MUST NOT contain a colon (':'). When "
        "using this algorithm, the first colon to appear in "
        "InstanceID MUST appear between <OrgID> and <LocalID>. \n"
        "\n"
        "<LocalID> is chosen by the business entity and SHOULD not be "
        "re-used to identify different underlying (real-world)
elements. "
        "If the above 'preferred' algorithm is not used, the defining "
        "entity MUST assure that the resultant InstanceID is not "
        "re-used across any InstanceIDs produced by this or other "
        "providers for this instance's NameSpace.") ]
  string InstanceID;

    [Description (
        "Describes the current status of this BatchJobGroup, based "
        "on the statuses of the jobs contained in the Group. The "
        "values of GroupStatus are more general than those of the "
        "individual jobs. The possible values are: \n"
        " 0 = Unknown \n"
        " 1 = Other \n"
        " 2 = All jobs pending \n"
        " 3 = Jobs in mixed states \n"
        " 4 = All jobs completed \n"
        "Additional information may be specified in the Group"
        "StatusDescription property."),
        ValueMap {"0", "1", "2", "3", "4"},
        Values {"Unknown", "Other", "All Jobs Pending",
            "Jobs in Mixed States", "All Jobs Completed"},
        ModelCorrespondence {"CIM_BatchJob.BatchJobStatus",
            "CIM_BatchJobGroup.GroupStatusDescription"} ]
    uint16 GroupStatus;

    [Description (
        "GroupStatusDescription provides additional information "
        "regarding the GroupStatus property."),
        ModelCorrespondence {"CIM_BatchJobGroup.GroupStatus"} ]
    string GroupStatusDescription;
};


// ==================================================================
// BatchSAP
// ==================================================================
[Experimental, Version ("2.9.0"), Description (
    "The ServiceAccessPoint for accessing a BatchService. The "
    "relationship between the AccessPoint and the Service is "
    "described by instantiating the ServiceAccessBySAP association.") ]
class CIM_BatchSAP : CIM_ServiceAccessPoint {

    [Description (
```

```
        "Specifies the batch submission procotols that this "
        "AccessPoint uses. Note that each entry of this array is "
        "related to the corresponding entry in the BatchProtocolInfo "
        "array that is located at the same index."),
        ArrayType ("Indexed"),
        ValueMap {"0", "1", "2"},
        Values {"Unknown", "Other", "Local"},
        ModelCorrespondence {"CIM_BatchSAP.BatchProtocolInfo"} ]
    uint16 BatchProtocol[];

    [Description (
        "Provides clarifying or additional information about the "
        "protocols supported by this AccessPoint. Note, each entry "
        "of this array is related to the corresponding entry in the "
        "BatchProtocol array that is located at the same index."),
        ArrayType ("Indexed"),
        ModelCorrespondence {"CIM_BatchSAP.BatchProtocol"} ]
    string BatchProtocolInfo[];
};


// ===================================================================
// BatchService
// ===================================================================
[Experimental, Version ("2.9.0"), Description (
    "The Service that provides support for processing BatchJob "
    "requests. The basic assumption of this model is that a "
    "BatchService accepts a BatchJob for processing, via its "
    "BatchSAP. The Job is then placed on a BatchQueue (indicated by "
    "the QueueForBatchService association). A System takes "
    "Jobs from Queues (indicated by the ExecutionTarget "
    "association) and processes them.") ]
class CIM_BatchService : CIM_Service {

    [Key, Description (
        "Within the scope of the instantiating Namespace, InstanceID "
        "opaquely and uniquely identifies an instance of this class. "
        "In order to ensure uniqueness within the NameSpace, the "
        "value of InstanceID SHOULD be constructed using the "
        "following 'preferred' algorithm: \n"
        "   <OrgID>:<LocalID> \n"
        "Where <OrgID> and <LocalID> are separated by a colon ':', "
        "and where <OrgID> MUST include a copyrighted, trademarked "
        "or otherwise unique name that is owned by the business entity
"
        "creating/defining the InstanceID, or is a registered ID that "
        "is assigned to the business entity by a recognized global "
        "authority. (This is similar to the <Schema Name>_<Class Name>
"
        "structure of Schema class names.) In addition, to ensure "
        "uniqueness <OrgID> MUST NOT contain a colon (':'). When "
        "using this algorithm, the first colon to appear in "
        "InstanceID MUST appear between <OrgID> and <LocalID>. \n"
        "\n"
        "<LocalID> is chosen by the business entity and SHOULD not be "
        "re-used to identify different underlying (real-world)
elements. "
```

```
        "If the above 'preferred' algorithm is not used, the defining "
        "entity MUST assure that the resultant InstanceID is not "
        "re-used across any InstanceIDs produced by this or other "
        "providers for this instance's NameSpace.") ]
  string InstanceID;

};


// ===================================================================
// QueueForBatchService
// ===================================================================
[Association, Experimental, Version ("2.8.0"), Description (
    "This association indicates that a BatchService utilizes a "
    "particular BatchQueue, to hold jobs submitted to the Service.") ]
class CIM_QueueForBatchService : CIM_Dependency {

    [Override ("Antecedent"), Description (
        "The BatchQueue that the Service utilizes.") ]
    CIM_BatchQueue REF Antecedent;

    [Override ("Dependent"), Description (
        "The BatchService that puts BatchJobs on the Queue.") ]
    CIM_BatchService REF Dependent;

    [Description (
        "Indicates that the BatchService can place jobs on the Queue.")
]
    boolean QueueAcceptingFromService;
};


// ===================================================================
// ExecutionCandidate
// ===================================================================
[Association, Experimental, Version ("2.8.0"), Description (
    "ExecutionCandidate represents the association between "
    "a System and a Job, describing that the System is "
    "capable of running the job.") ]
class CIM_ExecutionCandidate : CIM_Dependency {

    [Override("Antecedent"), Description (
        "The System that is capable of running the Job.") ]
    CIM_System REF Antecedent;

    [Override("Dependent"), Description (
        "The Job that can be run.") ]
    CIM_Job REF Dependent;
};


// ===================================================================
// ExecutionTarget
// ===================================================================
[Association, Experimental, Version ("2.8.0"), Description (
    "ExecutionTarget represents the association between "
    "a System and a Job, describing that the System is running "
```

```
    "the job.") ]
class CIM_ExecutionTarget : CIM_Dependency {

    [Override("Antecedent"), Description (
        "The System that is running the Job.") ]
    CIM_System REF Antecedent;

    [Override("Dependent"), Description (
        "The Job that is run.") ]
    CIM_Job REF Dependent;
};


// ==================================================================
// OwningCollectionElement
// ==================================================================
[Association, Experimental, Version ("2.8.0"), Description (
    "OwningCollectionElement represents an association between a
    "Collection and the ManagedElement responsible for the creation of
    "the Collection.  This association may not be possible, given that
"
    "the execution of collections of jobs can move between systems and
"
    "that the lifecycle of the creating entity may not persist for the
"
    "total duration of the collection.  However, this can be very "
    "useful information when available. ") ]
class CIM_OwningCollectionElement {

    [Key, Max(1), Description (
        "The ManagedElement responsible for the creation of the "
        "Collection.") ]
    CIM_ManagedElement REF OwningElement;

    [Key, Description (
        "The Collection created by the ManagedElement.") ]
    CIM_Collection REF OwnedElement;
};


// ==================================================================
// ExecutionRequirement
// ==================================================================
[Association, Experimental, Version ("2.8.0"), Description (
    "ExecutionRequirement represents the association of a job "
    "submitted to one or more ConcreteCollection defined by the "
    "Service for execution.") ]
class CIM_ExecutionCandidate : CIM_Dependency {

    [Override("Antecedent"), Description (
        "The ConcreteCollection that is required to run the Job.") ]
    CIM_ConcreteCollection REF Antecedent;

    [Override("Dependent"), Description (
        "The Job that can be run.") ]
    CIM_Job REF Dependent;
};
```

```
// ================================================================
// end of file
// ================================================================
```

6. XML

An XML representation of JSIM as an XSD is being developed. This will be made available as soon as it is approved by the CGS work group, but we do not intend to propose it as a standard. It will provide a test framework for getting data into and out of the JSIM model and integrating with schedulers, meta-schedulers, and indexing services.

7. Security Considerations

This specification defines the model and schema for job submission. While the interactions of job submission must be secured, the security details are outside the scope of this specification. Instead, it is assumed that security is addressed in specifications that define how this model and schema are bound to specific communication protocols (such as [CIMOPS] or [OGSI]) and programming environments.

8. Editor Information

Ellen Stokes
11400 Burnet Rd
Austin, TX 78758
Phone: +1 512 838 0552
Email: stokese@us.ibm.com

Lawrence Flon
USC/ISI
4676 Admiralty Way
Marina Del Rey, CA 90292
Email: flon@isi.edu

9. Acknowledgements

We are grateful to numerous colleagues for discussions on the topics covered in this document, in particular (in alphabetical order, with apologies to anybody we've missed): Peter Gietz, Carl Kesselman, Andreas Maier, Viktor Mihajlovski, Heidi Neumann, Tom Roney, and Andrea Westerinen.

10. References

[RFC2119]
Bradner, Scott, "Key Words for use in RFCs to Indicate Requirement Levels", RFC 2119, http://www.ietf.org/rfc/rfc2119.txt

[CIMspec]
DMTF, CIM Specification V2.2, http://www.dmtf.org/standards/documents/CIM/DSP0004.pdf

[CIM2.8prelim]
DMTF CIM Schema, Version 2.8 Preliminary, http://www.dmtf.org/standards/cim_schema_v28.php

[CIMOPS]
DMTF, CIM Operations over HTTP, http://www.dmtf.org/standards/documents/CIM/DSP0200.zip

[OGSI]
GGF, OGSI Draft Recommendation V1.5, http://prdownloads.sourceforge.net/ggf/GFD-R-P.15.pdf?download