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.

Copyright Notice

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. It includes a UML diagram of the classes associated with job submission, the managed object format (MOF) for those classes, and an XML representation of the UML.

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 CIM 2.8 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.

We 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 schema classes) are: BatchService, BatchSAP, BatchJob, BatchJobGroup, QueueForBatchService, ExecutionCandidate, ExecutionTarget, ExecutionRequirement, and OwningCollectionElement.[1]

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 more than one instance reported for 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 the CIM ConcreteJob and CIM Job classes, 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, respectively.

It is thus possible to locate all of the BatchServices 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 its associated JobQueue and BatchService.

Related aspects of the model include ConcreteCollection, which may be instantiated to define a collection of CIM ComputerSystems (nodes) or other elements such as CIM Processors, etc. This facilitates the control by a job scheduler (BatchService) and subsequent allocation of a BatchJob to a group of computer systems in a cluster, a group of processors on a single computer, etc. The control of the collection is identified by the association, OwningCollectionElement, and the allocation is identified via the association, ExecutionRequirement.

A Job and JobSettingData can include recovery action data.

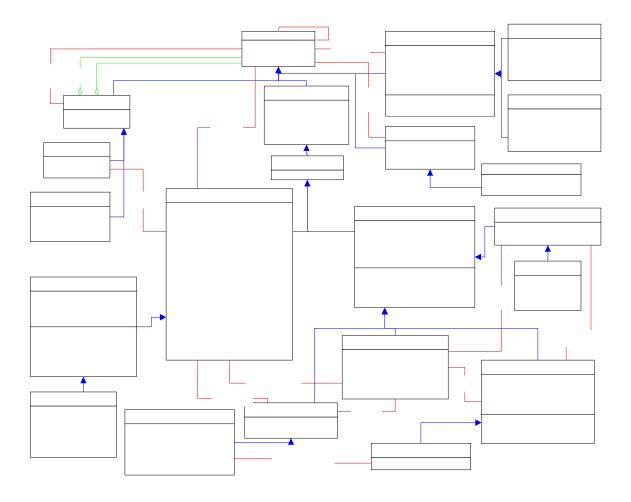[1] These elements will be submitted to DMTF for inclusion in CIM V2.9.

BatchSAP specifies the batch submission protocols (i.e., "Service Access Points") that BatchService uses. The relationship between the protocols/access points and the service 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. (MOF is a textual rendering of UML, defined in the CIM Specification [CIMspec] published by the DMTF.) Section 6 offers a corresponding XML description.

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 and be managed 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 **bold**.

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 CIM_ConcreteJob (and thence Job), with the specific meaning that, while ConcreteJob refers to any job that may be queued or executing on a System, a BatchJob is on a batch processing queue and traceable in that fashion.

Contents:
- InstanceID (inherited from ConcreteJob, Required)
  An arbitrary string uniquely identifying this job.

- Name (inherited from ManagedSystemElement, Required)
  The label by which the object is known.

- JobState (inherited from ConcreteJob)
  Describes the current state of this BatchJob with respect to the BatchQueue and the system executing this job.

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

- 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. BatchJobGroups are related to the element that controls them, and to their members through the associations OwningCollectionElement and OrderedMemberOfCollection. Note that BatchJobGroup does not inherit from CIM_ConcreteCollection, since ConcreteCollection MUST NOT be subclassed. CIM_ConcreteCollection is only for instantiating general collections, adding no semantics to the abstract class, Collection.

Contents:
- InstanceID
  An opaque string uniquely identifying this job group.

- 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.

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 a Collection. In JSIM, the BatchService is owning/controlling one or more ConcreteCollections.

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

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.

```
// ===================================================================
// Title:  GGF Batch Job Information Model
// Filename:  JobSubmission.mof
// Version:  1.0
// Date:  17 May 2004
// ===================================================================
// Description: This file defines the classes to manage batch job
//    submission. It is written in the Distributed Management Task
//    Force's (www.dmtf.org) Managed Object Format (MOF), and is a
//    textual rendering of UML. All required classes (and superclasses)
//    are included.
//
//    The classes are a subset of the DMTF's Common Information Model
//    (CIM) v2.9 Preliminary. The standard CIM MOF files (where the
//    classes are normatively defined) are listed in the comments.
// ===================================================================
// Change Log
// Feb 25 2003 - Initial release of the GGF batch job MOF
// Feb 28 2003 - Added Key property to BatchJobGroup
//        Clarified Descriptions for BatchJob, BatchJob.BatchJobStatus
// Jun 11 2003 - Many changes based on review comments
//        Added ExecutionRequirement association
// Jun 26 2003 - Changes from GGF CGS WG session
// Aug 27 2003 - Modified to supplement CIM 2.8 Prelim
// Nov 13 2003  - Removed InstanceID from BatchService, Fixed
//        misc associations
// May 11 2004 - Updated to include all superclassing and the
//        CIM v2.9 classes
// May 12 2004 - Added qualifier definitions and CIM_System to compile
// May 17 2004 - Added the Write qualifier to 2 properties in BatchJob
//        and to 4 properties in JobQueue / Added the class,
//        CIM_JobSettingData, and its superclass
// ===================================================================

#pragma locale ("en_US")


// ===================================================================
// QUALIFIERS
// ===================================================================

Qualifier Abstract : boolean = false,
    Scope(class, association, indication),
    Flavor(Restricted);
Qualifier Aggregate: boolean = false,
    Scope (reference),
    Flavor(DisableOverride);
Qualifier Aggregation : boolean = false,
    Scope(association),
    Flavor(DisableOverride);
Qualifier ArrayType : string = "Bag",
    Scope(property, parameter),
    Flavor(DisableOverride);
Qualifier Association : boolean = false,
    Scope(association),
    Flavor(DisableOverride);
Qualifier Composition : boolean = false,
    Scope(association),
    Flavor(DisableOverride);
Qualifier Counter : boolean = false,
    Scope(property, method, parameter);
Qualifier Description : string = null,
    Scope(any),
    Flavor(Translatable);
Qualifier Experimental : boolean = false,
    Scope(any),
    Flavor(Restricted);
Qualifier Gauge : boolean = false,
    Scope(property, method, parameter);
Qualifier In : boolean = true,
```

```
    Scope(parameter),
    Flavor(DisableOverride);
Qualifier Indication : boolean = false,
    Scope(class, indication),
    Flavor(DisableOverride);
Qualifier Key : boolean = false,
    Scope(property, reference),
    Flavor(DisableOverride);
Qualifier MappingStrings : string[],
    Scope(any);
Qualifier Max : uint32 = null,
    Scope(reference);
Qualifier MaxLen : uint32 = null,
    Scope(property, method, parameter);
Qualifier MaxValue : sint64 = null,
    Scope(property, method, parameter);
Qualifier Min : uint32 = 0,
    Scope(reference);
Qualifier MinLen : uint32 = 0,
    Scope(property, method, parameter);
Qualifier MinValue : sint64 = null,
    Scope(property, method, parameter);
Qualifier ModelCorrespondence : string[],
    Scope(any);
Qualifier Out : boolean = false,
    Scope(parameter),
    Flavor(DisableOverride);
Qualifier Override : string = null,
    Scope(property, method, reference),
    Flavor(Restricted);
Qualifier Propagated : string = null,
    Scope(property),
    Flavor(DisableOverride);
Qualifier Provider : string = null,
    Scope(any);
Qualifier Read : boolean = true,
    Scope(property);
Qualifier Required : boolean = false,
    Scope(property, reference, parameter),
    Flavor(DisableOverride);
Qualifier Static : boolean = false,
    Scope(method, property),
    Flavor(DisableOverride);
Qualifier Terminal : boolean = FALSE,
    Scope(class, association, indication);
Qualifier Units : string = null,
    Scope(property, method, parameter),
    Flavor(Translatable);
Qualifier ValueMap : string[],
    Scope(property, method, parameter);
Qualifier Values : string[],
    Scope(property, method, parameter),
    Flavor(Translatable);
Qualifier Version : string = null,
    Scope(class, association, indication),
    Flavor(Restricted, Translatable);
Qualifier Weak : boolean = false,
    Scope(reference),
    Flavor(DisableOverride, ToSubclass);
Qualifier Write : boolean = false,
    Scope(property);


// ==================================================================
// CORE CIM CLASSES (Top of the object hierarchies)
// ==================================================================


// ==================================================================
// ManagedElement (from Core_CoreElements.mof)
// ==================================================================
```

```
   [Abstract, Version ( "2.7.0" ), Description (
      "ManagedElement is an abstract class that provides a common "
      "superclass (or top of the inheritance tree) for the "
      "non-association classes in the CIM Schema.")]
class CIM_ManagedElement {

   [Description (
      "The Caption property is a short textual description (one- "
      "line string) of the object."),
      MaxLen ( 64 )]
   string Caption;

   [Description (
      "The Description property provides a textual description of "
      "the object.")]
   string Description;

   [Description (
      "A user-friendly name for the object. This property allows "
      "each instance to define a user-friendly name IN ADDITION TO "
      "its key properties/identity data, and description "
      "information. \n"
      "Note that ManagedSystemElement's Name property is also "
      "defined as a user-friendly name. But, it is often "
      "subclassed to be a Key. It is not reasonable that the same "
      "property can convey both identity and a user friendly name, "
      "without inconsistencies. Where Name exists and is not a Key "
      "(such as for instances of LogicalDevice), the same "
      "information MAY be present in both the Name and ElementName "
      "properties.")]
   string ElementName;
};


// ====================================================================
// Dependency (from Core_CoreElements.mof)
// ====================================================================
   [Association, Abstract, Version ( "2.6.0" ), Description (
      "CIM_Dependency is a generic association used to establish "
      "dependency relationships between ManagedElements.")]
class CIM_Dependency {

   [Key, Description (
      "Antecedent represents the independent object in this "
      "association.")]
   CIM_ManagedElement REF Antecedent;

   [Key, Description (
      "Dependent represents the object dependent on the "
      "Antecedent.")]
   CIM_ManagedElement REF Dependent;
};


// ====================================================================
// HostedDependency (from Core_CoreElements.mof)
// ====================================================================
   [Association, Version ( "2.8.0" ), Description (
      "HostedDependency defines a ManagedElement in the context of "
      "another ManagedElement in which it resides.")]
class CIM_HostedDependency : CIM_Dependency {

   [Override ( "Antecedent" ), Max ( 1 ), Description (
      "The scoping ManagedElement.")]
   CIM_ManagedElement REF Antecedent;

   [Override ( "Dependent" ), Description (
      "The hosted ManagedElement.")]
   CIM_ManagedElement REF Dependent;
};
```

```
// ====================================================================
// ManagedSystemElement (from Core_CoreElements.mof)
// ====================================================================
   [Abstract, Version ( "2.8.0" ), Description (
      "CIM_ManagedSystemElement is the base class for the System "
      "Element hierarchy. Any distinguishable component of a System "
      "is a candidate for inclusion in this class. Examples of system "
      "components include: \n"
      "- software components such as application servers, databases, "
      "and applications \n"
      "- operating system components such as files, processes, and "
      "threads \n"
      "- device components such as disk drives, controllers, "
      "processors, and printers \n"
      "- physical components such as chips and cards.")]
class CIM_ManagedSystemElement : CIM_ManagedElement {

   [Description (
      "A datetime value indicating when the object was installed. "
      "A lack of a value does not indicate that the object is not "
      "installed."),
    MappingStrings { "MIF.DMTF|ComponentID|001.5" }]
   datetime InstallDate;

   [Description (
      "The Name property defines the label by which the object is "
      "known. When subclassed, the Name property can be overridden "
      "to be a Key property."),
    MaxLen ( 1024 )]
   string Name;

   [Description (
      "Indicates the current status(es) of the element. Various "
      "health and operational statuses are defined. Many of the "
      "enumeration's values are self- explanatory. However, a few "
      "are not and are described in more detail. \n"
      "\"Stressed\" indicates that the element is functioning, but "
      "needs attention. Examples of \"Stressed\" states are "
      "overload, overheated, etc. \n"
      "\"Predictive Failure\" indicates that an element is "
      "functioning nominally but predicting a failure in the near "
      "future. \n"
      "\"In Service\" describes an element being configured, "
      "maintained, cleaned, or otherwise administered. \n"
      "\"No Contact\" indicates that the monitoring system has "
      "knowledge of this element, but has never been able to "
      "establish communications with it. \n"
      "\"Lost Communication\" indicates that the ManagedSystem "
      "Element is known to exist and has been contacted "
      "successfully in the past, but is currently unreachable. \n"
      "\"Stopped\" and \"Aborted\" are similar, although the "
      "former implies a clean and orderly stop, while the latter "
      "implies an abrupt stop where the element's state and "
      "configuration may need to be updated. \n"
      "\"Dormant\" indicates that the element is inactive or "
      "quiesced. \n"
      "\"Supporting Entity in Error\" describes that this element "
      "may be \"OK\" but that another element, on which it is "
      "dependent, is in error. An example is a network service or "
      "endpoint that cannot function due to lower layer networking "
      "problems. \n"
      "\"Completed\" indicates the element has completed its "
      "operation. This value should be combined with either OK, "
      "Error, or Degraded so that a client can till if the "
      "complete operation passed (Completed with OK), and failure "
      "(Completed with Error). Completed with Degraded would imply "
      "the operation finished, but did not complete OK or report "
      "an error. \n"
```

```
        "\"Power Mode\" indicates the element has additional power "
        "model information contained in the Associated "
        "PowerManagementService association."),
      ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
        "10", "11", "12", "13", "14", "15", "16", "17", "18", "..",
        "0x8000.." },
      Values { "Unknown", "Other", "OK", "Degraded", "Stressed",
        "Predictive Failure", "Error", "Non-Recoverable Error",
        "Starting", "Stopping", "Stopped", "In Service",
        "No Contact", "Lost Communication", "Aborted", "Dormant",
        "Supporting Entity in Error", "Completed", "Power Mode",
        "DMTF Reserved", "Vendor Reserved" }, ArrayType ( "Indexed" ),
      ModelCorrespondence {
        "CIM_ManagedSystemElement.StatusDescriptions" }]
   uint16 OperationalStatus[];

     [Description (
        "Strings describing the various OperationalStatus array "
        "values. For example, if \"Stopping\" is the value assigned "
        "to OperationalStatus, then this property may contain an "
        "explanation as to why an object is being stopped. Note that "
        "entries in this array are correlated with those at the same "
        "array index in OperationalStatus."),
      ArrayType ( "Indexed" ),
      ModelCorrespondence {
        "CIM_ManagedSystemElement.OperationalStatus" }]
   string StatusDescriptions[];
};


// ===================================================================
// LogicalElement (from Core_CoreElements.mof)
// ===================================================================
   [Abstract, Version ( "2.6.0" ), Description (
      "CIM_LogicalElement is a base class for all the components of a "
      "System that represent abstract system components, such as "
      "Files, Processes, or LogicalDevices.")]
class CIM_LogicalElement : CIM_ManagedSystemElement {
};


// ===================================================================
// Job (from Core_CoreElements.mof)
// ===================================================================
   [Abstract, Version ( "2.8.0" ), Description (
      "A Job is a LogicalElement representing an executing unit of "
      "work, such as a script or a print job. A Job is distinct from "
      "a Process in that a Job can be scheduled, queued, and its "
      "execution is not limited to a single system.")]
class CIM_Job : CIM_LogicalElement {

     [Description (
        "A free form string representing the Job's status. The "
        "primary status is reflected in the inherited "
        "OperationalStatus property. JobStatus provides additional, "
        "implementation-specific details."),
      ModelCorrespondence {
        "CIM_ManagedSystemElement.OperationalStatus" }]
   string JobStatus;

     [Description (
        "The time that the Job was submitted to execute. A value of "
        "all zeroes indicates that the owning element is not capable "
        "of reporting a date and time.")]
   datetime TimeSubmitted;

     [Description (
        "The time that the Job was actually started. This may be "
        "represented by an actual date and time, or by an interval "
        "relative to the time that this property is requested. Note "
```

```
        "that this property is also present in the "
        "JobProcessingStatistics class. This is necessary to capture "
        "the processing information for recurring Jobs, since only "
        "the 'last' run time can be stored in this single-valued "
        "property.")]
    datetime StartTime;

      [Description (
        "The time interval that the Job has been executing or the "
        "total execution time if the Job is complete. Note that this "
        "property is also present in the JobProcessingStatistics "
        "class. This is necessary to capture the processing "
        "information for recurring Jobs, since only the 'last' run "
        "time can be stored in this single-valued property.")]
    datetime ElapsedTime;

      [Write, Description (
        "Number of times that the Job should be run. A value of 1 "
        "indicates that the Job is NOT recurring, while any non-zero "
        "value indicates a limit to the number of times that the Job "
        "will recur. Zero indicates that there is no limit to the "
        "number of times that the Job can be processed, but that it "
        "is terminated either AFTER the UntilTime, or by manual "
        "intervention. By default, a Job is processed once.")]
    uint32 JobRunTimes = 1;

      [Write, Description (
        "The month during which the Job should be processed. Specify "
        "0 for January, 1 for February, and so on."),
       ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
        "10", "11" },
       Values { "January", "February", "March", "April", "May", "June",
        "July", "August", "September", "October", "November",
        "December" },
       ModelCorrespondence { "CIM_Job.RunDay", "CIM_Job.RunDayOfWeek",
        "CIM_Job.RunStartInterval" }]
    uint8 RunMonth;

      [Write, Description (
        "The day in the month on which the Job should be processed. "
        "There are two different interpretations for this property, "
        "depending on the value of DayOfWeek. In one case, RunDay "
        "defines the day-in-month on which the Job is processed. "
        "This interpretation is used when the DayOfWeek is 0. A "
        "positive or negative integer indicates whether the RunDay "
        "should be calculated from the beginning or the end of the "
        "month. For example, 5 indicates the fifth day in RunMonth "
        "and -1 indicates the last day in the RunMonth. \n"
        "\n"
        "When RunDayOfWeek is not 0, RunDay is the day-in-month on "
        "which the Job is processed, defined in conjunction with "
        "RunDayOfWeek. For example, if RunDay is 15 and RunDayOfWeek "
        "is Saturday, then the Job is processed on the first "
        "Saturday on or AFTER the 15th day in the RunMonth (e.g., "
        "the third Saturday in the month). If RunDay is 20 and "
        "RunDayOfWeek is -Saturday, then this indicates the first "
        "Saturday on or BEFORE the 20th day in the RunMonth. If "
        "RunDay is -1 and RunDayOfWeek is -Sunday, then this "
        "indicates the last Sunday in the RunMonth."),
       MinValue ( -31 ), MaxValue ( 31 ),
       ModelCorrespondence { "CIM_Job.RunMonth",
        "CIM_Job.RunDayOfWeek", "CIM_Job.RunStartInterval" }]
    sint8 RunDay;

      [Write, Description (
        "Positive or negative integer used in conjunction with "
        "RunDay to indicate the day of the week on which the Job is "
        "processed. RunDayOfWeek is set to 0 to indicate an exact "
        "day of the month, such as March 1. A positive integer "
        "(representing Sunday, Monday, ..., Saturday) means that the "
```

```
          "day of week is found on or AFTER the specified RunDay. A "
          "negative integer (representing -Sunday, -Monday, ..., "
          "-Saturday) means that the day of week is found on or BEFORE "
          "the RunDay."),
       ValueMap { "-7", "-6", "-5", "-4", "-3", "-2", "-1", "0", "1",
          "2", "3", "4", "5", "6", "7" },
       Values { "-Saturday", "-Friday", "-Thursday", "-Wednesday",
          "-Tuesday", "-Monday", "-Sunday", "ExactDayOfMonth",
          "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
          "Friday", "Saturday" },
       ModelCorrespondence { "CIM_Job.RunMonth", "CIM_Job.RunDay",
          "CIM_Job.RunStartInterval" }]
    sint8 RunDayOfWeek;

       [Write, Description (
          "The time interval after midnight when the Job should be "
          "processed. For example, \n"
          "00000000020000.000000:000 \n"
          "indicates that the Job should be run on or after two "
          "o'clock, local time or UTC time (distinguished using the "
          "LocalOrUtcTime property."),
       ModelCorrespondence { "CIM_Job.RunMonth", "CIM_Job.RunDay",
          "CIM_Job.RunDayOfWeek", "CIM_Job.RunStartInterval" }]
    datetime RunStartInterval;

       [Write, Description (
          "This property indicates whether the times represented in "
          "the RunStartInterval and UntilTime properties represent "
          "local times or UTC times. Time values are synchronized "
          "worldwide by using the enumeration value 2, \"UTC Time\"."),
       ValueMap { "1", "2" },
       Values { "Local Time", "UTC Time" }]
    uint16 LocalOrUtcTime;

       [Write, Description (
          "The time after which the Job is invalid or should be "
          "stopped. This may be represented by an actual date and "
          "time, or by an interval relative to the time that this "
          "property is requested. A value of all nines indicates that "
          "the Job can run indefinitely."),
       ModelCorrespondence { "CIM_Job.LocalOrUtcTime" }]
    datetime UntilTime;

       [Write, Description (
          "User to be notified upon the Job completion or failure.")]
    string Notify;

       [Description (
          "The User that submitted the Job or the Service/method "
          "name/etc. that caused the job to be created."),
       ModelCorrespondence { "CIM_OwningJobElement" }]
    string Owner;

       [Write, Description (
          "Indicates the urgency or importance of execution of the "
          "Job. The lower the number, the higher the priority. Note "
          "that this property is also present in the "
          "JobProcessingStatistics class. This is necessary to capture "
          "the setting information that would influence a Job's "
          "results.")]
    uint32 Priority;

       [Description (
          "The percentage of the job that has completed at the time "
          "that this value is requested. Note that this property is "
          "also present in the JobProcessingStatistics class. This is "
          "necessary to capture the processing information for "
          "recurring Jobs, since only the 'last' run data can be "
          "stored in this single-valued property. \n"
          "Note that the value 101 is undefined and will be not be "
```

```
        "allowed in the next major revision of the specification."),
      Units ( "Percent" ), MinValue ( 0 ), MaxValue ( 101 )]
   uint16 PercentComplete;

      [Write, Description (
         "Indicates whether or not the job should be automatically "
         "deleted upon completion. Note that the 'completion' of a "
         "recurring job is defined by its JobRunTimes or UntilTime "
         "properties, OR when the Job is terminated by manual "
         "intervention. If this property is set to false and the job "
         "completes, then the extrinsic method DeleteInstance MUST be "
         "used to delete the job versus updating this property.")]
   boolean DeleteOnCompletion;

      [Description (
         "A vendor specific error code. This MUST be set to zero if "
         "the Job completed without error."),
       ModelCorrespondence { "CIM_Job.ErrorDescription" }]
   uint16 ErrorCode;

      [Description (
         "A free form string containing the vendor error description."),
       ModelCorrespondence { "CIM_Job.ErrorCode" }]
   string ErrorDescription;

      [Description (
         "Describes the recovery action to be taken for an "
         "unsuccessfully run Job. The possible values are: \n"
         "0 = \"Unknown\", meaning it is unknown as to what recovery "
         "action to take \n"
         "1 = \"Other\", indicating that the recovery action will be "
         "specified in the property, OtherRecoveryAction \n"
         "2 = \"Do Not Continue\", meaning stop the execution of the "
         "job and appropriately update its status \n"
         "3 = \"Continue With Next Job\", meaning continue with the "
         "next job in the queue \n"
         "4 = \"Re-run Job\", indicating that the job should be "
         "re-run \n"
         "5 = \"Run Recovery Job\", meaning run the Job associated "
         "using the RecoveryJob relationship. Note that the recovery "
         "Job MUST already be on the queue from which it will run."),
       ValueMap { "0", "1", "2", "3", "4", "5" },
       Values { "Unknown", "Other", "Do Not Continue",
         "Continue With Next Job", "Re-run Job", "Run Recovery Job" },
       ModelCorrespondence { "CIM_Job.OtherRecoveryAction" }]
   uint16 RecoveryAction;

      [Description (
         "A string describing the recovery action when the instance's "
         "RecoveryAction property is 1 (\"Other\")."),
       ModelCorrespondence { "CIM_Job.RecoveryAction" }]
   string OtherRecoveryAction;
};


// ===================================================================
// ConcreteJob (from Core_CoreElements.mof)
// ===================================================================
   [Version ( "2.8.0" ), Description (
      "A concrete version of Job. This class represents a generic and "
      "instantiable unit of work, such as a batch or a print job.")]
class CIM_ConcreteJob : CIM_Job {

      [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"
                       "<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. \n"
                       "For DMTF defined instances, the 'preferred' algorithm MUST "
                       "be used with the <OrgID> set to 'CIM'.")]
          string InstanceID;

             [Required, Override ( "Name" ), Description (
                       "The user friendly name for this instance of Job. In "
                       "addition, the user friendly name can be used as a property "
                       "for a search or query. (Note: Name does not have to be "
                       "unique within a namespace.)")]
          string Name;

             [Description (
                       "JobState is an integer enumeration that indicates the "
                       "operational state of a Job. It can also indicate "
                       "transitions between these states, for example, 'Shutting "
                       "Down' and 'Starting'. Following is a brief description of "
                       "the states: \n"
                       "New (2) indicates that the job has never been started. \n"
                       "Starting (3) indicates that the job is moving from the "
                       "'New', 'Suspended', or 'Service' states into the 'Running' "
                       "state. \n"
                       "Running (4) indicates that the Job is running. \n"
                       "Suspended (5) indicates that the Job is stopped, but may be "
                       "restarted in a seamless manner. \n"
                       "Shutting Down (6) indicates the job is moving to a "
                       "'Completed', 'Terminated', or 'Killed' state. \n"
                       "Completed (7) indicates that the job has completed "
                       "normally. \n"
                       "Terminated (8) indicates that the job has been stopped by a "
                       "'Terminate' state change request. The job and all its "
                       "underlying processes are ended and may be restarted (this "
                       "is job-specific) only as a new job. \n"
                       "Killed (9) indicates that the job has been stopped by a "
                       "'Kill' state change request. Underlying processes may have "
                       "been left running and cleanup may be required to free up "
                       "resources. \n"
                       "Exception (10) indicates that the Job is in an abnormal "
                       "state that may be indicative of an error condition. Actual "
                       "status may be surfaced though job-specific objects. \n"
                       "Service (11) indicates that the Job is in a vendor-specific "
                       "state that supports problem discovery and/or resolution."),
                    ValueMap { "2", "3", "4", "5", "6", "7", "8", "9", "10", "11",
                       "12..32767", "32768..65535" },
                    Values { "New", "Starting", "Running", "Suspended",
                       "Shutting Down", "Completed", "Terminated", "Killed",
                       "Exception", "Service", "DMTF Reserved", "Vendor Reserved" }]
          uint16 JobState;

             [Description (
                       "The date/time when the Job's State last changed. If the "
                       "state of the Job has not changed and this property is "
                       "populated, then it MUST be set to a 0 interval value. If a "
                       "state change was requested, but rejected or not yet "
```

```
            "processed, the property MUST NOT be updated.")]
    datetime TimeOfLastStateChange;

        [Description (
            "Requests that the job's state be changed to the value "
            "specified in the RequestedState parameter. Invoking the "
            "RequestStateChange method multiple times could result in "
            "earlier requests being overwritten/lost. \n"
            "If 0 is returned, then the task completed successfully. Any "
            "other return code indicates an error condition."),
        ValueMap { "0", "1", "2", "3", "4", "5", "6", "..", "4096",
            "4097", "4098", "4099", "4100..32767", "32768..65535" },
        Values { "Completed with No Error", "Not Supported",
            "Unknown/Unspecified Error",
            "Can NOT complete within Timeout Period", "Failed",
            "Invalid Parameter", "In Use", "DMTF Reserved",
            "Method Parameters Checked - Transition Started",
            "Invalid State Transition",
            "Use of Timeout Parameter Not Supported", "Busy",
            "Method Reserved", "Vendor Specific" }]
    uint32 RequestStateChange(
        [IN, Description (
            "Changes the state of a job. \n"
            "Start (2) changes the state to 'Running'. \n"
            "Suspend (3) stops the job temporarily. The intention is "
            "to subsequently restart the job with 'Start'. It may be "
            "possible to enter the 'Service' state while suspended - "
            "this is job-specific. \n"
            "Terminate (4) stops the job cleanly, saving data, "
            "preserving state, and shutting down all underlying "
            "processes in an orderly manner. \n"
            "Kill (5) terminates the job immediately with no "
            "requirement to save data or preserve state. Service (6) "
            "puts the job into a vendor-specific service state. The "
            "job may be able to be restarted."),
        ValueMap { "2", "3", "4", "5", "6", "7..32767",
            "32768..65535" },
        Values { "Start", "Suspend", "Terminate", "Kill", "Service",
            "DMTF Reserved", "Vendor Reserved" }]
    uint16 RequestedState,
        [IN, Description (
            "A timeout period that specifies the maximum amount of "
            "time that the client expects the transition to the new "
            "state to take. The interval format MUST be used to "
            "specify the TimeoutPeriod. A value of 0 or a null "
            "parameter indicates that the client has no time "
            "requirements for the transition. \n"
            "If this property does not contain 0 or null and the "
            "implementation doesn't support this parameter. A return "
            "code of 'Use Of Timeout Parameter Not Supported' MUST be "
            "returned.")]
    datetime TimeoutPeriod);
};


// ===================================================================
// EnabledLogicalElement (from Core_CoreElements.mof)
// ===================================================================
    [Abstract, Version ( "2.8.0" ), Description (
        "This class extends LogicalElement to abstract the concept of "
        "an element that is enabled and disabled, such as a "
        "LogicalDevice or a ServiceAccessPoint.")]
class CIM_EnabledLogicalElement : CIM_LogicalElement {

        [Description (
            "EnabledState is an integer enumeration that indicates the "
            "enabled/disabled states of an element. It can also indicate "
            "the transitions between these requested states. For "
            "example, shutting down (value = 4) and starting (value=10) "
            "are transient states between enabled and disabled. The "
```

```
        "following text briefly summarizes the various "
        "enabled/disabled states: \n"
        "Enabled (2) indicates that the element is/could be "
        "executing commands, will process any queued commands, and "
        "queues new requests. \n"
        "Disabled (3) indicates that the element will not execute "
        "commands and will drop any new requests. \n"
        "Shutting Down (4) indicates that the element is in the "
        "process of going to a Disabled state. \n"
        "Not Applicable (5) indicates the element doesn't support "
        "being enabled/disabled. \n"
        "Enabled but Offline (6) indicates that the element may be "
        "completing commands, and will drop any new requests. \n"
        "Test (7) indicates that the element is in a test state. \n"
        "Deferred (8) indicates that the element may be completing "
        "commands, but will queue any new requests. \n"
        "Quiesce (9) indicates that the element is enabled but in a "
        "restricted mode. The element's behavior is similar to the "
        "Enabled state, but it only processes a restricted set of "
        "commands. All other requests are queued. \n"
        "Starting (10) indicates that the element is in the process "
        "of going to an Enabled state. New requests are queued."),
     ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
        "10", "11..32767", "32768..65535" },
     Values { "Unknown", "Other", "Enabled", "Disabled",
        "Shutting Down", "Not Applicable", "Enabled but Offline",
        "In Test", "Deferred", "Quiesce", "Starting",
        "DMTF Reserved", "Vendor Reserved" },
     ModelCorrespondence {
        "CIM_EnabledLogicalElement.OtherEnabledState" }]
  uint16 EnabledState = 5;

     [Description (
        "A string describing the element's enabled/disabled state "
        "when the EnabledState property is set to 1 (\"Other\"). "
        "This property MUST be set to NULL when EnabledState is any "
        "value other than 1."),
     ModelCorrespondence { "CIM_EnabledLogicalElement.EnabledState" }]
  string OtherEnabledState;

     [Description (
        "RequestedState is an integer enumeration indicating the "
        "last requested or desired state for the element. The actual "
        "state of the element is represented by EnabledState. This "
        "property is provided to compare the last requested and "
        "current enabled/disabled states. Note that when "
        "EnabledState is set to 5 (\"Not Applicable\"), then this "
        "property has no meaning. By default, the element's "
        "RequestedState is 5 (\"No Change\"). Refer to the "
        "EnabledState's property Description for explanations of the "
        "values in the RequestedState enumeration. \n"
        "\n"
        "It should be noted that there are two new values in "
        "RequestedState that build on the statuses of EnabledState. "
        "These are \"Reboot\" (10) and \"Reset\" (11). The former, "
        "Reboot, refers to doing a \"Shut Down\" and then moving to "
        "an \"Enabled\" state. The latter, Reset, indicates that the "
        "element is first \"Disabled\" and then \"Enabled\". The "
        "distinction between requesting \"Shut Down\" and "
        "\"Disabled\" should also be noted. The former, Shut Down, "
        "requests an orderly transition to the Disabled state, and "
        "MAY involve removing power, to completely erase any "
        "existing state. The latter, the Disabled state, requests an "
        "immediate disabling of the element, such that it will not "
        "execute or accept any commands or processing requests. \n"
        "\n"
        "This property is set as the result of a method invocation "
        "(such as Start or StopService on CIM_Service), or may be "
        "overridden and defined as WRITEable in a subclass. The "
        "method approach is considered superior to a WRITEable "
```

```
      "property, since it allows an explicit invocation of the "
      "operation and the return of a result code."),
    ValueMap { "2", "3", "4", "5", "6", "7", "8", "9", "10", "11",
      "..", "32768..65535" },
    Values { "Enabled", "Disabled", "Shut Down", "No Change",
      "Offline", "Test", "Deferred", "Quiesce", "Reboot", "Reset",
      "DMTF Reserved", "Vendor Reserved" },
    ModelCorrespondence { "CIM_EnabledLogicalElement.EnabledState" }]
uint16 RequestedState = 5;

  [Write, Description (
      "An enumerated value indicating an administrator's "
      "default/startup configuration for an element's Enabled "
      "State. By default, the element is \"Enabled\" (value=2)."),
    ValueMap { "2", "3", "5", "6", "7", "8..32767", "32768..65535" },
    Values { "Enabled", "Disabled", "Not Applicable",
      "Enabled but Offline", "No Default", "DMTF Reserved",
      "Vendor Reserved" }]
uint16 EnabledDefault = 2;

  [Description (
      "The date/time when the element's EnabledState last changed. "
      "If the state of the element has not changed and this "
      "property is populated, then it MUST be set to a 0 interval "
      "value. If a state change was requested, but rejected or not "
      "yet processed, the property MUST NOT be updated.")]
datetime TimeOfLastStateChange;

  [Description (
      "Requests that the element's state be changed to the value "
      "specified in the RequestedState parameter. When the "
      "requested state change takes place, the element's "
      "EnabledState and RequestedState will be the same. Invoking "
      "the RequestStateChange method multiple times could result "
      "in earlier requests being overwritten/lost. \n"
      "If 0 is returned, then the task completed successfully and "
      "the use of ConcreteJob was not required. If 4096 (0x1000) "
      "is returned, then the task will take some time to complete, "
      "ConcreteJob will be created, and its reference returned in "
      "the output parameter Job. Any other return code indicates "
      "an error condition."),
    ValueMap { "0", "1", "2", "3", "4", "5", "6", "..", "4096",
      "4097", "4098", "4099", "4100..32767", "32768..65535" },
    Values { "Completed with No Error", "Not Supported",
      "Unknown/Unspecified Error",
      "Can NOT complete within Timeout Period", "Failed",
      "Invalid Parameter", "In Use", "DMTF Reserved",
      "Method Parameters Checked - Job Started",
      "Invalid State Transition",
      "Use of Timeout Parameter Not Supported", "Busy",
      "Method Reserved", "Vendor Specific" },
    ModelCorrespondence {
      "CIM_EnabledLogicalElement.RequestedState" }]
uint32 RequestStateChange(
    [IN, Description (
      "The state requested for the element. This information "
      "will be placed into the instance's RequestedState "
      "property if the RequestStateChange method's return code "
      "is 0, 3, or 4096 (0x1000) - 'Completed with No Error', "
      "'Timeout', or 'Job Started' respectively. Refer to the "
      "description of the EnabledState and RequestedState "
      "properties for the detailed meanings of the "
      "RequestedState values."),
    ValueMap { "2", "3", "4", "6", "7", "8", "9", "10", "11",
      "..", "32768..65535" },
    Values { "Enabled", "Disabled", "Shut Down", "Offline",
      "Test", "Defer", "Quiesce", "Reboot", "Reset",
      "DMTF Reserved", "Vendor Reserved" },
    ModelCorrespondence {
      "CIM_EnabledLogicalElement.RequestedState" }]
```

```
    uint16 RequestedState,
      [IN ( false ), OUT, Description (
         "Reference to the job (may be null if task completed).")]
    CIM_ConcreteJob REF Job,
      [IN, Description (
         "A timeout period that specifies the maximum amount of "
         "time that the client expects the transition to the new "
         "state to take. The interval format MUST be used to "
         "specify the TimeoutPeriod. A value of 0 or a null "
         "parameter indicates that the client has no time "
         "requirements for the transition. \n"
         "If this property does not contain 0 or null and the "
         "implementation doesn't support this parameter. A return "
         "code of 'Use Of Timeout Parameter Not Supported' MUST be "
         "returned.")]
    datetime TimeoutPeriod);
};


// ====================================================================
// System (from Core_CoreElements.mof)
// ====================================================================
  [Abstract, Version ( "2.7.0" ), Description (
     "CIM_System represents an entity made up of component parts "
     "(defined by the SystemComponent relationship), that operates "
     "as a 'functional whole'. Systems are top level objects in the "
     "CIM hierarchy, requiring no scoping or weak relationships in "
     "order to exist and have context. It should be reasonable to "
     "uniquely name and manage a System at an enterprise level. For "
     "example, a ComputerSystem is a kind of System that can be "
     "uniquely named and independently managed in an enterprise. "
     "However, this is not true for the power supply (or the power "
     "supply sub-'system') within the computer. \n"
     "\n"
     "Although a System may be viewed as a Collection, this is not "
     "the correct model. A Collection is simply a 'bag' that 'holds' "
     "its members. A System is a higher level abstraction, built out "
     "of its individual components. It is more than a sum of its "
     "parts. Note that System is a subclass of EnabledLogicalElement "
     "which allows the entire abstraction to be functionally "
     "enabled/disabled - at a higher level than enabling/disabling "
     "its component parts.")]
class CIM_System : CIM_EnabledLogicalElement {

    [Key, Description (
       "CreationClassName indicates the name of the class or the "
       "subclass used in the creation of an instance. When used "
       "with the other key properties of this class, this property "
       "allows all instances of this class and its subclasses to be "
       "uniquely identified."),
      MaxLen ( 256 )]
  string CreationClassName;

    [Key, Override ( "Name" ), Description (
       "The inherited Name serves as key of a System instance in an "
       "enterprise environment."),
      MaxLen ( 256 )]
  string Name;

    [Description (
       "The System object and its derivatives are Top Level Objects "
       "of CIM. They provide the scope for numerous components. "
       "Having unique System keys is required. A heuristic can be "
       "defined in individual System subclasses to attempt to "
       "always generate the same System Name Key. The NameFormat "
       "property identifies how the System name was generated, "
       "using the subclass' heuristic."),
      MaxLen ( 64 )]
  string NameFormat;
```

```
  [Write, Description (
      "The name of the primary system owner. The system owner is "
      "the primary user of the system."),
   MaxLen ( 64 ),
   MappingStrings { "MIF.DMTF|General Information|001.3" }]
string PrimaryOwnerName;

  [Write, Description (
      "A string that provides information on how the primary "
      "system owner can be reached (e.g. phone number, email "
      "address, ...)."),
   MaxLen ( 256 ),
   MappingStrings { "MIF.DMTF|General Information|001.4" }]
string PrimaryOwnerContact;

  [Write, Description (
      "An array (bag) of strings that specify the administrator "
      "-defined roles this System plays in the managed "
      "environment. Examples might be 'Building 8 print server' or "
      "'Boise user directories'. A single system may perform "
      "multiple roles. \n"
      "Note that instrumentation's view of a System's 'roles' is "
      "defined by instantiating a specific subclass of System and/ "
      "or by properties in a subclass. For example, a "
      "ComputerSystem's purpose is defined using the Dedicated and "
      "OtherDedicatedDescription properties.")]
string Roles[];
};

// =================================================================
// COLLECTION, SETTING AND STATISTICS SUPERCLASSES
// =================================================================

// =================================================================
// Collection (from Core_Collection.mof)
// =================================================================
  [Abstract, Version ( "2.6.0" ), Description (
      "Collection is an abstract class that provides a common "
      "superclass for data elements that represent collections of "
      "ManagedElements and its subclasses.")]
class CIM_Collection : CIM_ManagedElement {
};

// =================================================================
// MemberOfCollection (from Core_Collection.mof)
// =================================================================
  [Association, Aggregation, Version ( "2.6.0" ), Description (
      "CIM_MemberOfCollection is an aggregation used to establish "
      "membership of ManagedElements in a Collection.")]
class CIM_MemberOfCollection {

  [Key, Aggregate, Description (
      "The Collection that aggregates members.")]
  CIM_Collection REF Collection;

  [Key, Description (
      "The aggregated member of the Collection.")]
  CIM_ManagedElement REF Member;
};

// =================================================================
// OrderedMemberOfCollection (from Core_Collection.mof)
// =================================================================
  [Association, Aggregation, Version ( "2.7.0" ), Description (
      "CIM_OrderedMemberOfCollection is an aggregation used to "
      "establish an ordered membership of ManagedElements in a "
      "Collection.")]
class CIM_OrderedMemberOfCollection : CIM_MemberOfCollection {

  [Description (
```

```
        "AssignedSequence is an unsigned integer 'n' that indicates "
        "the relative position of members within a Collection. When "
        "'n' is a positive integer, it indicates a place in the "
        "sequence of members, with smaller integers indicating "
        "earlier positions in the sequence. The special value '0' "
        "indicates 'don't care'. If two or more members have the "
        "same non-zero sequence number, then the ordering between "
        "those members is irrelavent, but they must all be ordered "
        "at the appropriate place in the overall sequence. \n"
        "\n"
        "A series of examples will make ordering of members clearer: "
        "\nIf all members have the same sequence number, \n"
        "regardless of whether it is '0' or non-zero, any \n"
        "order is acceptable. \n"
        "o The values: \n"
        "1:MEMBER A \n"
        "2:MEMBER B \n"
        "1:MEMBER C \n"
        "3:MEMBER D \n"
        "indicate two acceptable orders: A,C,B,D or C,A,B,D, \n"
        "since A and C can be ordered in either sequence, but \n"
        "only at the '1' position. \n"
        "\n"
        "Note that the non-zero sequence numbers need not start with "
        "'1', and they need not be consecutive. All that matters is "
        "their relative magnitude.")]
    uint64 AssignedSequence;
};

// ====================================================================
// ConcreteCollection (from Core_Collection.mof)
// ====================================================================
    [Version ( "2.8.0" ), Description (
        "A concrete version of Collection. This class represents a "
        "generic and instantiable collection, such as a group of "
        "ComputerSystems (e.g., a pool of hosts available for executing "
        "Jobs). It is defined as a concrete subclass of the abstract "
        "CIM_Collection class, to be used in place of many specific "
        "subclasses that add no semantics. Note that when defining "
        "additional semantics for Collection that this class MUST NOT "
        "be subclassed. Specific semantics continue to be defined as "
        "subclasses of the abstract CIM_Collection. ConcreteCollection "
        "is limited in its use as a concrete form of a general "
        "collection or bag. It was deemed more prudent to create this "
        "concrete subclass than to change Collection from an abstract "
        "to a concrete class.")]
class CIM_ConcreteCollection : 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;
};


// ====================================================================
// OwningCollectionElement (from Core_Collection.mof)
// ====================================================================
    [Association, Experimental, Version ( "2.8.1000" ), Description (
        "OwningCollectionElement represents an association between a "
        "Collection and the ManagedElement responsible for the control/ "
        "ownership of the Collection.")]
class CIM_OwningCollectionElement {

    [Key, Max ( 1 ), Description (
        "The ManagedElement acting as the 'owner'/responsible for "
        "the control of the Collection.")]
    CIM_ManagedElement REF OwningElement;

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


// ====================================================================
// SettingData (from Core_Settings.mof)
// ====================================================================
    [Abstract, Version ( "2.8.1000" ), Description (
        "The SettingData class represents configuration-related and "
        "operational parameters for one or more ManagedElement(s). A "
        "ManagedElement may have multiple SettingData objects "
        "associated with it. The current operational values for an "
        "Element's parameters are reflected by properties in the "
        "Element itself or by properties in its associations. These "
        "properties do not have to be the same values present in the "
        "SettingData object. For example, a modem may have a "
        "SettingData baud rate of 56Kb/sec but be operating at "
        "19.2Kb/sec.")]
class CIM_SettingData : CIM_ManagedElement {

    [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"
        "<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. \n"
        "For DMTF defined instances, the 'preferred' algorithm MUST "
        "be used with the <OrgID> set to 'CIM'.")]
    string InstanceID;

    [Required, Override ( "ElementName" ), Description (
```

```
      "The user friendly name for this instance of SettingData. In "
      "addition, the user friendly name can be used as a index "
      "property for a search of query. (Note: Name does not have "
      "to be unique within a namespace.)")]
  string ElementName;
};


// ====================================================================
// ElementSettingData (from Core_Settings.mof)
// ====================================================================
  [Association, Version ( "2.8.1000" ), Description (
      "ElementSettingData represents the association between "
      "ManagedElements and applicable setting data. This association "
      "also describes whether this is a default or current setting.")]
class CIM_ElementSettingData {

    [Key, Description (
        "The managed element.")]
  CIM_ManagedElement REF ManagedElement;

    [Key, Description (
        "The SettingData object associated with the element.")]
  CIM_SettingData REF SettingData;

    [Description (
        "An enumerated integer indicating that the referenced "
        "setting is a default setting for the element, or that this "
        "information is unknown."),
     ValueMap { "0", "1", "2" },
     Values { "Unknown", "Is Default", "Is Not Default" }]
  uint16 IsDefault;

    [Description (
        "An enumerated integer indicating that the referenced "
        "setting is currently being used in the operation of the "
        "element, or that this information is unknown."),
     ValueMap { "0", "1", "2" },
     Values { "Unknown", "Is Current", "Is Not Current" }]
  uint16 IsCurrent;
};


// ====================================================================
// StatisticalData (from Core_Statistics.mof)
// ====================================================================
  [Abstract, Version ( "2.8.0" ), Description (
      "CIM_StatisticalData is a root class for any arbitrary "
      "collection of statistical data and/or metrics applicable to "
      "one or more ManagedElements. These statistics MUST represent "
      "the most recent observations and MUST NOT be provided if "
      "irrelevant or stale. Note that this class uses a simplified "
      "naming/identity algorithm as compared to "
      "CIM_StatisticalInformation.")]
class CIM_StatisticalData : CIM_ManagedElement {

    [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"
            "<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. \n"
            "For DMTF defined instances, the 'preferred' algorithm MUST "
            "be used with the <OrgID> set to 'CIM'.")]
    string InstanceID;

      [Required, Override ( "ElementName" ), Description (
            "The user friendly name for this instance of "
            "StatisticalData. In addition, the user friendly name can be "
            "used as a index property for a search of query. (Note: Name "
            "does not have to be unique within a namespace.)")]
    string ElementName;

      [Description (
            "The time, relative to managed element where the statistic "
            "was collected, when the first measurement was taken. If the "
            "statistic is reset, the StartStatisticTime is the time when "
            "the reset was performed.")]
    datetime StartStatisticTime;

      [Description (
            "The time the most recent measurement was taken, relative to "
            "the managed element where the statistic was collected.")]
    datetime StatisticTime;

      [Description (
            "Some statistics are sampled at consistent time intervals. "
            "This property provides the sample interval so that client "
            "applications can determine the minimum time that new "
            "statistics should be pulled. If the statistics are not "
            "sampled at consistent time intervals, this property must be "
            "set to a zero time interval.")]
    datetime SampleInterval = "00000000000000.000000:000";

      [Description (
            "Method to reset one or more of the instance's statistics. "
            "The method takes one parameter as input - an array of "
            "strings indicating which statistics to reset. If all the "
            "statistics in the instance should be reset, the first "
            "element of the array MUST be set to \"All\" or \"ALL\". If "
            "one or more individual statistics should be reset, the "
            "corresponding property names are entered into the elements "
            "of the array. \n"
            "The method returns 0 if successful, 1 if not supported, and "
            "any other value if an error occurred. A method is specified "
            "so that the StatisticalInformation's provider/ "
            "instrumentation, which calculates the statistics, can reset "
            "its internal processing, counters, etc. \n"
            "In a subclass, the set of possible return codes could be "
            "specified, using a ValueMap qualifier on the method. The "
            "strings to which the ValueMap contents are 'translated' may "
            "also be specified in the subclass as a Values array "
            "qualifier.")]
    uint32 ResetSelectedStats (
        [IN, Description (
            "Array of strings indicating which statistics to reset.")]
        string SelectedStatistics[]);
};

// ====================================================================
// ElementStatisticalData (from Core_Statistics.mof)
// ====================================================================
    [Association, Version ( "2.7.0" ), Description (
        "CIM_ElementStatisticalData is an association that relates a "
```

```
            "ManagedElement to its StatisticalData. Note that the "
            "cardinality of the ManagedElement reference is Min(1), Max(1). "
            "This cardinality mandates the instantiation of the "
            "ElementStatisticalData association for the referenced instance "
            "of CIM_StatisticalData. ElementStatisticalData describes the "
            "existence requirements and context for the "
            "CIM_StatisticalData, relative to a specific ManagedElement.")]
class CIM_ElementStatisticalData {

    [Key, Min ( 1 ), Max ( 1 ), Description (
        "The ManagedElement for which statistical or metric data is "
        "defined.")]
    CIM_ManagedElement REF ManagedElement;

    [Key, Description (
        "The statistic information/object.")]
    CIM_StatisticalData REF Stats;
};

// ====================================================================
// SERVICE AND ACCESS POINT CIM CLASSES
// ====================================================================

// ====================================================================
// Service (from Core_Service.mof)
// ====================================================================
    [Abstract, Version ( "2.7.0" ), Description (
        "A CIM_Service is a Logical Element that contains the "
        "information necessary to represent and manage the "
        "functionality provided by a Device and/or SoftwareFeature. A "
        "Service is a general-purpose object to configure and manage "
        "the implementation of functionality. It is not the "
        "functionality itself.")]
class CIM_Service : CIM_EnabledLogicalElement {

    [Key, Propagated ( "CIM_System.CreationClassName" ),
     Description (
        "The scoping System's CreationClassName."),
     MaxLen ( 256 )]
    string SystemCreationClassName;

    [Key, Propagated ( "CIM_System.Name" ), Description (
        "The scoping System's Name."),
     MaxLen ( 256 )]
    string SystemName;

    [Key, Description (
        "CreationClassName indicates the name of the class or the "
        "subclass used in the creation of an instance. When used "
        "with the other key properties of this class, this property "
        "allows all instances of this class and its subclasses to be "
        "uniquely identified."),
     MaxLen ( 256 )]
    string CreationClassName;

    [Key, Override ( "Name" ), Description (
        "The Name property uniquely identifies the Service and "
        "provides an indication of the functionality that is "
        "managed. This functionality is described in more detail in "
        "the object's Description property."),
     MaxLen ( 256 )]
    string Name;

    [Write, Description (
        "The name of the primary owner for the service, if one is "
        "defined. The primary owner is the initial support contact "
        "for the Service."),
     MaxLen ( 64 ),
     MappingStrings { "MIF.DMTF|General Information|001.3" }]
    string PrimaryOwnerName;
```

```
    [Write, Description (
        "A string that provides information on how the primary owner "
        "of the Service can be reached (e.g. phone number, email "
        "address, ...)."),
     MaxLen ( 256 ),
     MappingStrings { "MIF.DMTF|General Information|001.4" }]
  string PrimaryOwnerContact;

    [Description (
        "Started is a boolean indicating whether the Service has "
        "been started (TRUE), or stopped (FALSE).")]
  boolean Started;

    [Description (
        "The StartService method places the Service in the started "
        "state. Note that this method's function overlaps with the "
        "RequestedState property. RequestedState was added to the "
        "model to maintain a record (i.e., a persisted value) of the "
        "last state request. Invoking the StartService method should "
        "set the RequestedState property appropriately. The method "
        "returns an integer value of 0 if the Service was "
        "successfully started, 1 if the request is not supported and "
        "any other number to indicate an error. In a subclass, the "
        "set of possible return codes could be specified, using a "
        "ValueMap qualifier on the method. The strings to which the "
        "ValueMap contents are 'translated' may also be specified in "
        "the subclass as a Values array qualifier. \n"
        "\n"
        "Note that this method overlaps in semantics with the "
        "RequestStateChange method inherited from "
        "EnabledLogicalElement. This method is maintained since it "
        "has been widely implemented, and its simple 'start' "
        "semantics are convenient to use.")]
  uint32 StartService();

    [Description (
        "The StopService method places the Service in the stopped "
        "state. Note that this method's function overlaps with the "
        "RequestedState property. RequestedState was added to the "
        "model to maintain a record (i.e., a persisted value) of the "
        "last state request. Invoking the StopService method should "
        "set the RequestedState property appropriately. The method "
        "returns an integer value of 0 if the Service was "
        "successfully stopped, 1 if the request is not supported and "
        "any other number to indicate an error. In a subclass, the "
        "set of possible return codes could be specified, using a "
        "ValueMap qualifier on the method. The strings to which the "
        "ValueMap contents are 'translated' may also be specified in "
        "the subclass as a Values array qualifier. \n"
        "\n"
        "Note that this method overlaps in semantics with the "
        "RequestStateChange method inherited from "
        "EnabledLogicalElement. This method is maintained since it "
        "has been widely implemented, and its simple 'stop' "
        "semantics are convenient to use.")]
  uint32 StopService();
};

// ====================================================================
// HostedService (from Core_Service.mof)
// ====================================================================
  [Association, Version ( "2.8.0" ), Description (
      "CIM_HostedService is an association between a Service and the "
      "System on which the functionality resides. The cardinality of "
      "this association is 1-to-many. A System may host many "
      "Services. Services are weak with respect to their hosting "
      "System. Heuristic: A Service is hosted on the System where the "
      "LogicalDevices or SoftwareFeatures that implement the Service "
      "are located. The model does not represent Services hosted "
```

```
       "across multiple systems. This is modeled as an "
       "ApplicationSystem that acts as an aggregation point for "
       "Services, that are each located on a single host.")]
class CIM_HostedService : CIM_HostedDependency {

    [Override ( "Antecedent" ), Min ( 1 ), Max ( 1 ), Description (
        "The hosting System.")]
  CIM_System REF Antecedent;

    [Override ( "Dependent" ), Weak, Description (
        "The Service hosted on the System.")]
  CIM_Service REF Dependent;
};

// ===================================================================
// ServiceAccessPoint (from Core_Service.mof)
// ===================================================================
  [Abstract, Version ( "2.6.0" ), Description (
      "CIM_ServiceAccessPoint represents the ability to utilize or "
      "invoke a Service. Access points represent that a Service is "
      "made available to other entities for use.")]
class CIM_ServiceAccessPoint : CIM_EnabledLogicalElement {

    [Key, Propagated ( "CIM_System.CreationClassName" ),
     Description (
        "The scoping System's CreationClassName."),
     MaxLen ( 256 )]
  string SystemCreationClassName;

    [Key, Propagated ( "CIM_System.Name" ), Description (
        "The scoping System's Name."),
     MaxLen ( 256 )]
  string SystemName;

    [Key, Description (
        "CreationClassName indicates the name of the class or the "
        "subclass used in the creation of an instance. When used "
        "with the other key properties of this class, this property "
        "allows all instances of this class and its subclasses to be "
        "uniquely identified."),
     MaxLen ( 256 )]
  string CreationClassName;

    [Key, Override ( "Name" ), Description (
        "The Name property uniquely identifies the "
        "ServiceAccessPoint and provides an indication of the "
        "functionality that is managed. This functionality is "
        "described in more detail in the object's Description "
        "property."),
     MaxLen ( 256 )]
  string Name;
};

// ===================================================================
// HostedAccessPoint (from Core_Service.mof)
// ===================================================================
  [Association, Version ( "2.8.0" ), Description (
      "CIM_HostedAccessPoint is an association between a Service "
      "AccessPoint and the System on which it is provided. The "
      "cardinality of this association is 1-to-many and is weak with "
      "respect to the System. Each System may host many "
      "ServiceAccessPoints. Heuristic: If the implementation of the "
      "ServiceAccessPoint is modeled, it must be implemented by a "
      "Device or SoftwareFeature that is part of the System hosting "
      "the ServiceAccessPoint.")]
class CIM_HostedAccessPoint : CIM_HostedDependency {

    [Override ( "Antecedent" ), Min ( 1 ), Max ( 1 ), Description (
        "The hosting System.")]
  CIM_System REF Antecedent;
```

```
    [Override ( "Dependent" ), Weak, Description (
        "The SAP(s) that are hosted on this System.")]
  CIM_ServiceAccessPoint REF Dependent;
};

// ================================================================
// ServiceAccessBySAP (from Core_Service.mof)
// ================================================================
  [Association, Version ( "2.6.0" ), Description (
      "CIM_ServiceAccessBySAP is an association that identifies the "
      "access points for a Service. For example, a printer may be "
      "accessed by Netware, MacIntosh or Windows ServiceAccess "
      "Points, potentially hosted on different Systems.")]
class CIM_ServiceAccessBySAP : CIM_Dependency {

    [Override ( "Antecedent" ), Description (
        "The Service.")]
  CIM_Service REF Antecedent;

    [Override ( "Dependent" ), Description (
        "An Access Point for a Service. Access points are dependent "
        "in this relationship since they have no function without a "
        "corresponding Service.")]
  CIM_ServiceAccessPoint REF Dependent;
};

// ================================================================
// JOB-RELATED CIM CLASSES
// ================================================================

// ================================================================
// BatchJob (from System_Processing.mof)
// ================================================================
  [Experimental, Version ( "2.8.1000" ), 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 JobSettingData instance.")]
class CIM_BatchJob : CIM_ConcreteJob {

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

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

    [Description (
        "Time when this BatchJob was completed. This value is only "
        "valid if the inherited JobState property has been set to "
        "\"Completed\" (value=7), \"Terminated\" (value=8), or "
        "\"Killed\" (value=9)."),
     ModelCorrespondence { "CIM_BatchJob.JobState" }]
  datetime TimeCompleted;

    [Write, 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;
};


// ====================================================================
// JobSettingData (from System_Processing.mof)
// ====================================================================
    [Experimental, Version ( "2.8.1000" ), Description (
        "JobSettingData represents job characteristics and scheduling "
        "information for a CIM_Job. The settings may be applied to a "
        "job handling Service, a JobQueue or other destination to which "
        "jobs are submitted, and/or to individual CIM_Jobs. The "
        "applicability of the settings are defined using the "
        "ElementSettingData relationship. Since the Settings may be "
        "defined in several contexts and applied to many types of "
        "elements, this class is defined as a subclass of SettingData, "
        "versus ScopedSettingData.")]
class CIM_JobSettingData : CIM_SettingData {

    [Write, Description (
        "Number of times that a Job should be run. A value of 1 "
        "indicates that the Job is NOT recurring, while any non-zero "
        "value indicates a limit to the number of times that the Job "
        "will recur. Zero indicates that there is no limit to the "
        "number of times that the Job can be processed, but that it "
        "is terminated either AFTER the UntilTime, or by manual "
        "intervention. By default, a Job is processed once."),
        ModelCorrespondence { "CIM_Job.JobRunTimes" }]
    uint32 JobRunTimes = 1;

    [Write, Description (
        "The month during which a Job should be processed. Specify 0 "
        "for January, 1 for February, and so on."),
        ValueMap { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
            "10", "11" },
        Values { "January", "February", "March", "April", "May", "June",
            "July", "August", "September", "October", "November",
            "December" },
        ModelCorrespondence { "CIM_Job.RunMonth",
            "CIM_JobSettingData.RunDay",
            "CIM_JobSettingData.RunDayOfWeek",
            "CIM_JobSettingData.RunStartInterval" }]
    uint8 RunMonth;

    [Write, Description (
        "The day in the month on which a Job should be processed. "
        "There are two different interpretations for this property, "
        "depending on the value of RunDayOfWeek. In one case, RunDay "
        "defines the day-in-month on which the Job is processed. "
        "This interpretation is used when the RunDayOfWeek is 0. A "
        "positive or negative integer indicates whether the RunDay "
        "should be calculated from the beginning or the end of the "
        "month. For example, 5 indicates the fifth day in RunMonth "
        "and -1 indicates the last day in the RunMonth. \n"
        "\n"
        "When RunDayOfWeek is not 0, RunDay is the day-in-month on "
        "which the Job is processed, defined in conjunction with "
        "RunDayOfWeek. For example, if RunDay is 15 and RunDayOfWeek "
        "is Saturday, then the Job is processed on the first "
        "Saturday on or AFTER the 15th day in the RunMonth (e.g., "
        "the third Saturday in the month). If RunDay is 20 and "
        "RunDayOfWeek is -Saturday, then this indicates the first "
        "Saturday on or BEFORE the 20th day in the RunMonth. If "
        "RunDay is -1 and RunDayOfWeek is -Sunday, then this "
        "indicates the last Sunday in the RunMonth."),
        MinValue ( -31 ), MaxValue ( 31 ),
        ModelCorrespondence { "CIM_Job.RunDay",
```

```
        "CIM_JobSettingData.RunMonth",
        "CIM_JobSettingData.RunDayOfWeek",
        "CIM_JobSettingData.RunStartInterval" }]
    sint8 RunDay;

      [Write, Description (
        "Positive or negative integer used in conjunction with "
        "RunDay to indicate the day of the week on which a Job is "
        "processed. RunDayOfWeek is set to 0 to indicate an exact "
        "day of the month, such as March 1. A positive integer "
        "(representing Sunday, Monday, ..., Saturday) means that the "
        "day of week is found on or AFTER the specified RunDay. A "
        "negative integer (representing -Sunday, -Monday, ..., "
        "-Saturday) means that the day of week is found on or BEFORE "
        "the RunDay."),
      ValueMap { "-7", "-6", "-5", "-4", "-3", "-2", "-1", "0", "1",
        "2", "3", "4", "5", "6", "7" },
      Values { "-Saturday", "-Friday", "-Thursday", "-Wednesday",
        "-Tuesday", "-Monday", "-Sunday", "ExactDayOfMonth",
        "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
        "Friday", "Saturday" },
      ModelCorrespondence { "CIM_Job.RunDayOfWeek",
        "CIM_JobSettingData.RunMonth", "CIM_JobSettingData.RunDay",
        "CIM_JobSettingData.RunStartInterval" }]
    sint8 RunDayOfWeek;

      [Write, Description (
        "The time interval after midnight when a Job should be "
        "processed. For example, \n"
        "00000000020000.000000:000 \n"
        "indicates that the Job should be run on or after two "
        "o'clock, local time or UTC time (distinguished using the "
        "LocalOrUtcTime property."),
      ModelCorrespondence { "CIM_Job.RunStartInterval",
        "CIM_JobSettingData.RunMonth", "CIM_JobSettingData.RunDay",
        "CIM_JobSettingData.RunDayOfWeek",
        "CIM_JobSettingData.LocalOrUtcTime" }]
    datetime RunStartInterval;

      [Write, Description (
        "This property indicates whether the times represented in "
        "the RunStartInterval and UntilTime properties represent "
        "local times or UTC times. Time values are synchronized "
        "worldwide by using the enumeration value 2, \"UTC Time\"."),
      ValueMap { "1", "2" },
      Values { "Local Time", "UTC Time" }]
    uint16 LocalOrUtcTime;

      [Write, Description (
        "The setting for a Job's UntilTime - the time after which an "
        "executing Job should be stopped or not be allowed to "
        "proceed. This may be represented by an actual date and "
        "time, or an interval relative to the ScheduledStartTime. A "
        "value of all nines indicates that the Job can run "
        "indefinitely."),
      ModelCorrespondence { "CIM_Job.UntilTime",
        "CIM_JobSettingData.LocalOrUtcTime" }]
    datetime UntilTime;

      [Write, Description (
        "The setting for a Job's Notify property - the user/email to "
        "be notified upon a scheduled Job's completion or failure."),
      ModelCorrespondence { "CIM_Job.Notify" }]
    string Notify;

      [Write, Description (
        "The setting for a Job's Owner property - the user, service "
        "method, etc. that defined this setting data and causes the "
        "Job to be created."),
      ModelCorrespondence { "CIM_Job.Owner" }]
```

```
    string Owner;

      [Write, Description (
          "Indicates the urgency or importance of execution of the "
          "Job, as specified by the user or defining entity. The "
          "actual Priority is found in the CIM_Job instance, "
          "associated via the ElementSettingData relationship. Unless "
          "otherwise specified, a lower number indicates a higher "
          "priority."),
        ModelCorrespondence { "CIM_Job.Priority" }]
    uint32 Priority;

      [Write, Description (
          "Indicates whether or not the Job should be automatically "
          "deleted upon completion. Note that the 'completion' of a "
          "recurring job is defined by its JobRunTimes or UntilTime "
          "properties, OR when the Job is terminated by manual "
          "intervention."),
        ModelCorrespondence { "CIM_Job.DeleteOnCompletion" }]
    boolean DeleteOnCompletion;

      [Description (
          "Describes the recovery setting if the Job does not "
          "unsuccessfully complete. The possible values are: \n"
          "0 = \"Unknown\", meaning it is unknown as to what recovery "
          "action to take \n"
          "1 = \"Other\", indicating that the recovery action will be "
          "specified in the property, OtherRecoveryAction \n"
          "2 = \"Do Not Continue\", meaning stop the execution of the "
          "job and appropriately update its status \n"
          "3 = \"Continue With Next Job\", meaning continue with the "
          "next job in the queue \n"
          "4 = \"Re-run Job\", indicating that the job should be "
          "re-run \n"
          "5 = \"Run Recovery Job\", meaning run the Job associated "
          "using the RecoveryJob relationship. Note that the recovery "
          "Job MUST already be on the queue from which it will run."),
        ValueMap { "0", "1", "2", "3", "4", "5" },
        Values { "Unknown", "Other", "Do Not Continue",
          "Continue With Next Job", "Re-run Job", "Run Recovery Job" },
        ModelCorrespondence { "CIM_JobSettingData.OtherRecoveryAction",
          "CIM_Job.RecoveryAction" }]
    uint16 RecoveryAction;

      [Description (
          "A string describing the recovery action when the instance's "
          "RecoveryAction property is 1 (\"Other\")."),
        ModelCorrespondence { "CIM_JobSettingData.RecoveryAction",
          "CIM_Job.OtherRecoveryAction" }]
    string OtherRecoveryAction;
};

// ==================================================================
// BatchJobGroup (from System_Processing.mof)
// ==================================================================
  [Experimental, Version ( "2.8.1000" ), 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, where the order is defined using the "
      "AssignedSequence property of the OrderedMemberOfCollection "
      "association. An individual BacthJob can belong to multiple "
      "BatchJobGroups. \n"
      "\n"
```

```
        "If possible, an instance of OwningCollectionElement SHOULD be "
        "created for this JobGroup. However, this association might not "
        "exist, given that collections of jobs can move between systems "
        "as they execute, and the lifecycle of the owning/controlling "
        "entity is not necessarily persisted for the total duration of "
        "the collection. However, this can be very useful information "
        "when available.")]
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 (from System_Processing.mof)
// ===================================================================
    [Experimental, Version ( "2.8.1000" ), 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."),
     ValueMap { "0", "1", "2" },
     Values { "Unknown", "Other", "Local" }, ArrayType ( "Indexed" ),
     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 (from System_Processing.mof)
// ====================================================================
  [Experimental, Version ( "2.8.1000" ), 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 {
};

// ====================================================================
// OwningJobElement (from System_Processing.mof)
// ====================================================================
  [Association, Version ( "2.7.0" ), Description (
      "OwningJobElement represents an association between a Job and "
      "the ManagedElement responsible for the creation of the Job. "
      "This association may not be possible, given that the execution "
      "of jobs can move between systems and that the lifecycle of the "
      "creating entity may not persist for the total duration of the "
      "job. However, this can be very useful information when "
      "available. This association defines a more specific 'owner' "
      "than is provided by the CIM_Job.Owner string."),
   ModelCorrespondence { "CIM_Job.Owner" }]
class CIM_OwningJobElement {

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

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

// ====================================================================
// JobDestination (from System_Processing.mof)
// ====================================================================
  [Version ( "2.7.0" ), Description (
      "JobDestination is an EnabledLogicalElement representing where "
      "a Job is submitted for processing. It can refer to a queue "
      "that contains zero or more Jobs, such as a print queue "
      "containing print Jobs. JobDestinations are hosted on Systems, "
      "similar to the way that Services are hosted on Systems. Also, "
      "they may be enabled/disabled similar to Services.")]
class CIM_JobDestination : CIM_EnabledLogicalElement {
```

```
    [Key, Propagated ( "CIM_System.CreationClassName" ),
     Description (
       "The scoping System's CreationClassName."),
      MaxLen ( 256 )]
  string SystemCreationClassName;

    [Key, Propagated ( "CIM_System.Name" ), Description (
       "The scoping System's Name."),
      MaxLen ( 256 )]
  string SystemName;

    [Key, Description (
       "CreationClassName indicates the name of the class or the "
       "subclass used in the creation of an instance. When used "
       "with the other key properties of this class, this property "
       "allows all instances of this class and its subclasses to be "
       "uniquely identified."),
      MaxLen ( 256 )]
  string CreationClassName;

    [Key, Override ( "Name" ), Description (
       "The inherited Name serves as key of a JobDestination "
       "instance in a System."),
      MaxLen ( 256 )]
  string Name;
};

// ===================================================================
// HostedJobDestination (from System_Processing.mof)
// ===================================================================
  [Association, Version ( "2.8.0" ), Description (
     "An association between a JobDestination and a System on which "
     "it resides. The cardinality of this association is 1-to-many. "
     "A System may host many Job queues. Job Destinations are weak "
     "with respect to their hosting System. Heuristic: A "
     "JobDestination is hosted on the System where the "
     "LogicalDevices, SoftwareFeatures or Services that implement/ "
     "provide the JobDestination are located.")]
class CIM_HostedJobDestination : CIM_HostedDependency {

    [Override ( "Antecedent" ), Min ( 1 ), Max ( 1 ), Description (
       "The hosting System.")]
  CIM_System REF Antecedent;

    [Override ( "Dependent" ), Weak, Description (
       "The JobDestination hosted on the System.")]
  CIM_JobDestination REF Dependent;
};

// ===================================================================
// JobDestinationJobs (from System_Processing.mof)
// ===================================================================
  [Association, Version ( "2.6.0" ), Description (
     "An association describing where a Job is submitted for "
     "processing, ie to which JobDestination.")]
class CIM_JobDestinationJobs : CIM_Dependency {

    [Override ( "Antecedent" ), Max ( 1 ), Description (
       "The JobDestination, possibly a queue.")]
  CIM_JobDestination REF Antecedent;

    [Override ( "Dependent" ), Description (
       "The Job that is in the Job queue/Destination.")]
  CIM_Job REF Dependent;
};

// ===================================================================
// JobQueue (from System_Processing.mof)
// ===================================================================
```

```
    [Experimental, Version ( "2.8.1000" ), Description (
        "Status and configuration of a JobQueue. Jobs are held on a "
        "queue, and can move from queue to queue until being processed.")]
class CIM_JobQueue : CIM_JobDestination {

    [Description (
        "Provides queue-specific status information, beyond the "
        "ManagedSystemElement.OperationalStatus property. This "
        "property further qualifies the status of the queue, or "
        "indicates a secondary condition (e.g. queue full)."),
     ValueMap { "0", "1", "2", "3" },
     Values { "Unknown", "Other", "No Additional Status",
        "Queue Full" },
     ModelCorrespondence { "CIM_JobQueue.QueueStatusInfo" }]
    uint16 QueueStatus;

    [Description (
        "Provides a textual explanation for the status of the queue. "
        "Information about both QueueStatus and the "
        "OperationalStatus property (inherited from "
        "ManagedSystemElement) may be described in QueueStatusInfo. "
        "Note, however, that OperationalStatus' Descriptions (if "
        "available) MUST be provided in the StatusDescriptions "
        "array, inherited from ManagedSystemElement. This is where "
        "management clients will expect to locate this information."),
     ModelCorrespondence { "CIM_JobQueue.QueueStatus" }]
    string QueueStatusInfo;

    [Write, Description (
        "Specifies the priority that will be assigned to a new Job "
        "being submitted to this queue, if the Job does not "
        "explicitly specify a priority. Note that lower numbers "
        "indicate higher priorities. The default value of 0xFFFF is "
        "the lowest priority.")]
    uint32 DefaultJobPriority = 65535;

    [Write, Description (
        "Specifies the maximum time (using a datetime interval "
        "format) that a Job can remain on this Queue before "
        "completing. If the MaxTimeOnQueue property is exceeded, "
        "then the Job SHOULD be removed from the Queue. A value of 0 "
        "(the default) indicates that there is no time limit.")]
    datetime MaxTimeOnQueue = "00000000000000.000000:000";

    [Write, Description (
        "Specifies the maximum number of Jobs that can reside on "
        "this Queue. A value of 0 (the default) indicates that there "
        "is no time limit.")]
    uint32 MaxJobsOnQueue = 0;

    [Write, Description (
        "Specifies the maximum number of milliseconds of CPU time "
        "that a Job queued to this destination can consume to "
        "complete its execution."),
     Units ( "MilliSeconds" )]
    uint32 MaxJobCPUTime;
};

// =====================================================================
// QueueStatisticalData (from System_Processing.mof)
// =====================================================================
    [Experimental, Version ( "2.8.1000" ), Description (
        "The statistical data of a JobQueue. An instance of this class "
        "is associated with a specific JobQueue using the inherited "
        "ElementStatisticalData relationship.")]
class CIM_QueueStatisticalData : CIM_StatisticalData {

    [Description (
        "The number of Jobs whose MaxTimeOnQueue was exceeded. This "
        "would cause the Job to be removed from the Queue, and not "
```

```
      "processed to completion. This count begins at the time that "
      "the Queue was started/initialized."),
   Counter]
  uint64 JobsMaxTimeExceeded;

   [Description (
      "Specifies the current number of Jobs that are in the "
      "running state, for the related Queue."),
   Gauge]
  uint32 RunningJobs;

   [Description (
      "Specifies the current number of Jobs that are waiting to be "
      "put in the running state, for the related Queue."),
   Gauge]
  uint32 WaitingJobs;
};

// ====================================================================
// QueueForBatchService (from System_Processing.mof)
// ====================================================================
  [Association, Experimental, Version ( "2.8.1000" ), Description (
      "This association indicates that a BatchService utilizes a "
      "particular JobQueue, to hold jobs submitted to the Service. At "
      "any point in time, the JobQueue can be full or otherwise not "
      "accepting jobs from the BatchService. This situation is "
      "indicated by the QueueAcceptingFromService Boolean property in "
      "this association.")]
class CIM_QueueForBatchService : CIM_Dependency {

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

// ====================================================================
// JobProcessingStatistics (from System_Processing.mof)
// ====================================================================
  [Experimental, Version ( "2.8.1000" ), Description (
      "This class records the completion statistics (such as "
      "ElapsedTime) and other information for its associated Job. The "
      "class will be extended in a future release to include "
      "statistics regarding resource utilization. Note that there may "
      "be many instances of this class if the Job is recurring (i.e., "
      "if its JobRunTimes property is not 1).")]
class CIM_JobProcessingStatistics : CIM_StatisticalData {

   [Description (
      "The time that this invocation of the Job was started. This "
      "SHOULD be represented by an actual date and time, and not "
      "in interval format."),
   ModelCorrespondence { "CIM_Job.StartTime" }]
  datetime StartTime;

   [Description (
      "The total execution time of the Job. This MUST be "
      "represented in interval format."),
   ModelCorrespondence { "CIM_Job.ElapsedTime" }]
  datetime ElapsedTime;

   [Description (
```

```
        "Indicates the urgency or importance of execution of the "
        "Job. The lower the number, the higher the priority."),
      ModelCorrespondence { "CIM_Job.Priority" }]
   uint32 Priority;

     [Description (
        "The percentage of the job that was completed at the time "
        "that the Job terminated. Note that this value will be less "
        "than 100 if the Job terminated prematurely. \n"
        "Note that the value 101 is undefined and will be not be "
        "allowed in the next major revision of the specification."),
      Units ( "Percent" ), MinValue ( 0 ), MaxValue ( 101 ),
      ModelCorrespondence { "CIM_Job.PercentComplete" }]
   uint16 PercentComplete;

     [Description (
        "Specifies the number of milliseconds of CPU that this job "
        "has used in its current (or last) run. When a job is "
        "actively running, this number will continue to change until "
        "the job has finished its execution, either successfully or "
        "unsuccessfully."),
      Units ( "MilliSeconds" ), Counter,
      ModelCorrespondence { "CIM_BatchJob.MaxCPUTime" }]
   uint32 CPUTimeUsed;
};


// ===================================================================
// ExecutionCandidate (from System_Processing.mof)
// ===================================================================
   [Association, Experimental, Version ( "2.8.1000" ), 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 (from System_Processing.mof)
// ===================================================================
   [Association, Experimental, Version ( "2.8.1000" ), 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;
};


// ===================================================================
// ExecutionRequirement (from System_Processing.mof)
// ===================================================================
   [Association, Experimental, Version ( "2.8.1000" ), Description (
      "ExecutionRequirement represents the restriction of a job to a "
      "specific set of nodes, processors, etc. that are aggregated "
      "into a ConcreteCollection.")]
class CIM_ExecutionRequirement : 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 XML 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 XML Schema are bound to specific communication protocols (such as [CIMOPS] or [OGSA]) 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

[CIM2.7final]
DMTF CIM Schema, Version 2.7 Final, http://www.dmtf.org/standards/cim_schema_v27.php

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

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

[CIMXML]
DMTF, Representation of CIM in XML, http://www.dmtf.org/standards/documents/CIM/DSP0201.pdf

[CIMLDAP]
DMTF, Guidelines for CIM-to-LDAP Directory Mapping, http://www.dmtf.org/standards/documents/DEN/DSP0100.pdf

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

11. Full Copyright and Intellectual Property Notices

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).