

CIM Specification 2.2 Errata Sheet: 01

In Section 2.1 "Definition of the Meta Schema", rule 21, item C (page 8)

Change the word "Schema" to "Namespace" in the second sentence so that it now reads:

Explicitly-defined Qualifier Names are unique within the defining Namespace.

In Section 2.5.2 "Standard Qualifiers" (page 15)

The following changes should be made to the table of standard qualifiers

GAUGE	FALSE	Property, Method, Parameter	BOOLEAN	Applicable only to integer types. Represents an integer which may increase or decrease in any order or magnitude. The value of a Gauge is capped at the implied limits of the effected property's datatype. If and while the information being modeled exceeds an implied limit, the value represented will instead be that limit; values do not wrap. For unsigned integers, the limits are zero (0) to 2^{n-1} , inclusive. For signed integers, the limits are $-(2^{(n-1)})$ to $2^{(n-1)}-1$, inclusive. N can be 8, 16, 32, or 64 depending on the
-------	-------	-----------------------------	---------	---

				datatype of the property to which the qualifier is applied.
KEY	FALSE	Property, Reference	BOOLEAN	<p>Indicates that the property is part of the namespace handle (see Section 5.3.1.2 for information about namespace handles). If more than one property has the KEY qualifier, then all such properties collectively form the key (a compound key).</p> <p>Usage Rule: Keys are written once at object instantiation and must not be modified thereafter. It does not make sense to apply a default value to a KEY-qualified property. Properties which are of an array type must not be qualified with KEY.</p>
OCTETSTRING	FALSE	Property, Parameter	BOOLEAN	<p>This qualifier is used to identify the qualified property or parameter as an octet string.</p> <p>When used in conjunction with an unsigned 8-bit integer (uint8) array, the OCTETSTRING qualifer indicates that the unsigned 8-bit integer array</p>

				<p>represents a single octet string.</p> <p>When used in conjunction with arrays of strings, the OCTETSTRING qualifier indicates that the qualified character strings are encoded textual conventions representing octetstrings. The text encoding of these binary values conforms to the following grammar: <code>Ox<&lt;0-9,A-F>&lt;0-9,A-F>>*</code>.</p> <p>In both cases, the first 4 octets of the octet string (8 hexadecimal digits in the text encoding) are the number of octets in the represented octet string with the length portion included in the octet count (e.g., "0x00000004" is the encoding of a 0-length octetstring).</p>
VERSION	NULL	Class, Association, Indication	STRING	Provides the major version number of the schema object. This is incremented when changes are made to the schema that alter the interface

In Section 3.1 "MOF Usage" (page 28)

Replace the existing paragraph with the following:

The managed object descriptions in a MOF specification can be validated against an active namespace (See Section 5). Such validation is typically implemented in an entity acting in the role of a Server. This section describes the behavior of an implementation when introducing a MOF specification into a namespace. Typically, such a process validates both the syntactic correctness of a MOF specification, as well as the semantic correctness of such a specification against a particular Implementation. In particular, MOF declarations must be ordered correctly with respect to the target implementation state. For example, if the specification references a class without defining it first, the reference is valid only if the server already has a definition of that class. A MOF specification can be validated for the syntactic correctness alone, in a component such as a MOF compiler

In Section 3.2 "Class Declarations" (page 28)

Replace the second paragraph with the following:

If the specification references a class without defining it first, the server must reject it as invalid if it does not already have a definition of that class.

In Section 3.3 "Instance Declarations" (page 28)

Replace the first paragraph with the following:

If the specification references a class without defining it first, the server must reject it as invalid if it does not already have a definition of that class.

In Appendix A "MOF Syntax Grammar Description" (page 72)

Replace the "metaElement" production (page 74) with the following:

```
metaElement    = CLASS | ASSOCIATION | INDICATION | QUALIFIER
                PROPERTY | REFERENCE | METHOD |
PARAMETER | ANY
```

In Appendix B "CIM Meta Schema" (page 77)

Add or replace the following qualifier definitions:

```
Qualifier OctetString : boolean = false, Scope(property, parameter);
Qualifier Version : string = null, Scope(class, association,
indication), Flavor(translatable);
```

In Appendix G "Guidelines" (page 95)

Replace the second paragraph of section G.1 with the following:

In the current release, CIM does not support octet strings as a separate data type. To map a single octet string (i.e., octets of binary data), it is recommended that the equivalent CIM property be defined as an array of unsigned 8-bit integers (uint8). The first four bytes of the array contain the length of the octet data: byte 0 is the most significant byte of the length; byte 3 is the least significant byte of the length. The octet data starts at byte 4. The OCTETSTRING qualifier may be used to indicate that the uint8 array conforms to this encoding.

Add the following paragraphs to the end of section G.1.

In the case where an array of octet strings must be mapped, since arrays of uint8 arrays are not supported, a textual convention encoding the binary information as hexadecimal digit characters (i.e., 0x<<0-9,A-F><0-9,A-F>>*) is used for each of the octet strings in the array. The number of octets in the octet string is encoded in the first 8 hexadecimal digits of the string with the most significant digits in the left-most characters of the string. The length count octets are included in the length count (i.e., "0x00000004" is the encoding of a 0-length octet string. The OCTETSTRING qualifier is used to qualify the string array.

Example usages of the Octetstring qualifier on a property follows:

```
[Description ("An octet string"), Octetstring]
uint8 Foo[];
[Description ("An array of octet strings"), Octetstring]
String Bar[];
```