Document Number: DSP1045

Date: 2009-07-14

Version: 1.0.0

Memory Resource Virtualization Profile

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Figures

Tables

Foreword

The *Memory Resource Virtualization Profile* (DSP1045) was prepared by the System Virtualization, Partitioning and Clustering Working Group of the DMTF.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability.

Editors:

- Michael Johanssen – IBM
- Ron Goering – IBM

Contributors:

- Gareth Bestor – IBM
- Ron Goering – IBM
- Daniel Hiltgen –VMware
- Ron Doyle – IBM
- Rene Schmidt – VMware Inc.
- Steffen Grarup – VMware Inc.
- Hemal Shah – Broadcom
- Fred Maciel – Hitachi Ltd.
- Lawrence Lamers – VMware Inc.
- Andreas Maier – IBM
- John Parchem – Microsoft Corporation
- George Ericson – EMC
- Oliver Benke – IBM
- John Leung – Intel Corporation
- James Fehlig – Novell
- Nihar Shah – Microsoft Corporation
- Shishir Pardikar – Citrix Systems Inc.
- Stephen Schmidt – IBM
- Mark Hapner – Sun Microsystems
- Dave Barrett – Emulex
- John Suit – Fortisphere
- Jeff Wheeler – Cisco
- Mark Johnson – IBM
- Carl Waldsburger – VMware Inc.

Introduction

The information in this specification should be sufficient for a provider or consumer of this data to identify unambiguously the classes, properties, methods, and values that shall be instantiated and manipulated to represent and manage the components described in this document. The target audience for this specification is implementers who are writing CIM-based providers or consumers of management interfaces that represent the components described in this document.

157 # Memory Resource Virtualization Profile

158 ## 1 Scope

159 This profile is a component DMTF management profile that extends the management capabilities of the
160 referencing profile by adding the support to represent and manage the allocation of memory to virtual
161 systems.

162 ## 2 Normative References

163 The following referenced documents are indispensable for the application of this document. For dated
164 references, only the edition cited applies. For undated references, the latest edition of the referenced
165 document (including any amendments) applies.

166 DMTF DSP0004, *CIM Infrastructure Specification* 2.5,
167 http://www.dmtf.org/standards/published_documents/DSP0004_2.5.pdf

168 DMTF DSP0200, *CIM Operations over HTTP* 1.3,
169 http://www.dmtf.org/standards/published_documents/DSP0200_1.3.pdf

170 DMTF DSP1001, *Management Profile Specification Usage Guide* 1.0,
171 http://www.dmtf.org/standards/published_documents/DSP1001_1.0.pdf

172 DMTF DSP1026, *System Memory Profile* 1.0,
173 http://www.dmtf.org/standards/published_documents/DSP1026_1.0.pdf

174 DMTF DSP1033, *Profile Registration Profile* 1.0,
175 http://www.dmtf.org/standards/published_documents/DSP1033_1.0.pdf

176 DMTF DSP1041, *Resource Allocation Profile* 1.1,
177 http://www.dmtf.org/standards/published_documents/DSP1041_1.1.pdf

178 DMTF DSP1043, *Allocation Capabilities Profile* 1.0,
179 http://www.dmtf.org/standards/published_documents/DSP1043_1.0.pdf

180 ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
181 http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype

182 ## 3 Terms and Definitions

183 For the purposes of this document, the following terms and definitions apply. For the purposes of this
184 document, the terms and definitions given in DSP1033 and DSP1001 also apply.

185 **3.1**
186 **can**
187 used for statements of possibility and capability, whether material, physical, or causal

188 **3.2**
189 **cannot**
190 used for statements of possibility and capability, whether material, physical, or causal

3.3
conditional
indicates requirements strictly to be followed in order to conform to the document and from which no deviation is permitted when the specified conditions are met

3.4
mandatory
indicates requirements strictly to be followed in order to conform to the document and from which no deviation is permitted

3.5
may
indicates a course of action permissible within the limits of the document

3.6
need not
indicates a course of action permissible within the limits of the document

3.7
optional
indicates a course of action permissible within the limits of the document

3.8
referencing profile
indicates a profile that owns the definition of this class and can include a reference to this profile in its "Related Profiles" table

3.9
shall
indicates requirements strictly to be followed in order to conform to the document and from which no deviation is permitted

3.10
shall not
indicates requirements strictly to be followed in order to conform to the document and from which no deviation is permitted

3.11
should
indicates that among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required

3.12
should not
indicates that a certain possibility or course of action is deprecated but not prohibited

3.13
unspecified
indicates that this profile does not define any constraints for the referenced CIM element

3.14
client
an application that exploits facilities specified by this profile

3.15
implementation
a set of CIM providers that realize the classes specified by this profile

3.16
this profile
this DMTF management profile – the *Memory Resource Virtualization Profile*

3.17
concrete memory resource pool
a resource pool that subdivides the capacity of its (primordial or concrete) parent resource pool

3.18
host memory
a contiguous extent of memory contained by the host system that may be allocated with either exclusive or shared access to a memory resource pool

3.19
host system
the scoping system that contains memory resources that may be allocated, virtualized, or both

3.20
memory composition
the aggregation of memory extents into an encompassing memory extent

3.21
memory resource allocation
the allocation of memory from a memory resource pool to a virtual system

3.22
memory resource allocation request
a request for a memory resource allocation

3.23
memory resource pool
a resource pool that represents memory available for memory resource allocation

3.24
memory resource pool configuration service
a configuration service that supports the addition or removal of host memory to or from a memory resource pool, and the creation or deletion of concrete subpools of a memory resource pool

3.25
primordial memory resource pool
a resource pool that aggregates memory available for or used by memory resource allocations

3.26
virtual computer system
the concept of a virtual system as applied to a computer system
Other common industry terms are *virtual machine*, *hosted computer*, *child partition*, *logical partition*, *domain*, *guest*, or *container*.

3.27
virtualization platform
virtualizing infrastructure provided by a host system that enables the deployment of virtual systems

3.28
virtual memory
the instantiation of allocated host memory that is exposed to a virtual system through a logical memory device; the result of a memory resource allocation based on a memory resource allocation request

NOTE: The definition of the term “virtual memory” is specialized from the term “virtual resource” defined in the *Resource Allocation Profile* (DSP1041) and deviates from common computer industry parlance.

4 Symbols and Abbreviated Terms

The following symbols and abbreviations are used in this document.

4.1
CIM
Common Information Model

4.2
CIMOM
CIM object manager

4.3
MCA
capabilities settings of systems and of memory resource pools

4.4
MMS
mutability settings of memory resource allocation requests or memory resource allocations

4.5
MRA
memory resource allocation

4.6
MRQ
memory resource allocation request

4.7
RASD
CIM_ResourceAllocationSettingData

4.8
VSSD
CIM_VirtualSystemSettingData

5 Synopsis

Profile Name: Memory Resource Virtualization

Version: 1.0.0

Organization: DMTF

CIM Schema Version: 2.22

Central Class: CIM_ResourcePool

Scoping Class: CIM_System

This profile is a component profile that defines the minimum object model needed to provide for the CIM representation and management of the virtualization of memory.

Table 1 lists DMTF management profiles on which this profile depends.

Table 1 – Related Profiles

Profile Name	Organization	Version	Relationship	Description
Resource Allocation	DMTF	1.1	Specializes	The abstract profile that describes the virtualization of resources See 7.2.
Allocation Capabilities	DMTF	1.0	Specializes	The abstract profile that describes capabilities for resource allocation See 7.3.
Profile Registration	DMTF	1.0	Mandatory	The profile that specifies registered profiles
System Memory	DMTF	1.0	Optional	The SMWG profile that specifies the management of system memory See 7.4.

6 Description (Informative)

This clause introduces the management domain addressed by the *Memory Resource Virtualization Profile*, and outlines the central modeling elements established for representation and control of the management domain.

6.1 General

In computer virtualization systems, virtual computer systems are composed of component virtual resources. This profile specifies the allocation and management of host computer system memory in support of virtual computer system memory. The memory described here would appear as “physical” memory to an operating system running in the virtual computer system. This profile is not intended to specify the management of virtual memory as virtualized by operating systems.

This profile applies the resource virtualization pattern defined in DSP1041 (*Resource Allocation Profile*) and the allocation capabilities pattern defined in DSP1043 (*Allocation Capabilities Profile*) to enable the

management of processor resources that are allocated to virtual systems. This profile defines additional CIM elements and constraints beyond those defined in the specialized profiles. Optionally implementations may implement DSP1026 (*System Memory Profile*) to represent host memory.

6.2 Memory Resource Virtualization Class Schema

Figure 1 shows the class schema of this profile. It outlines the elements that are referenced and in some cases further constrained by this profile, as well as the dependency relationships between elements of this profile and other profiles. For simplicity in diagrams, the prefix *CIM_* has been removed from class and association names. Inheritance relationships are shown only to the extent required in the context of this profile.

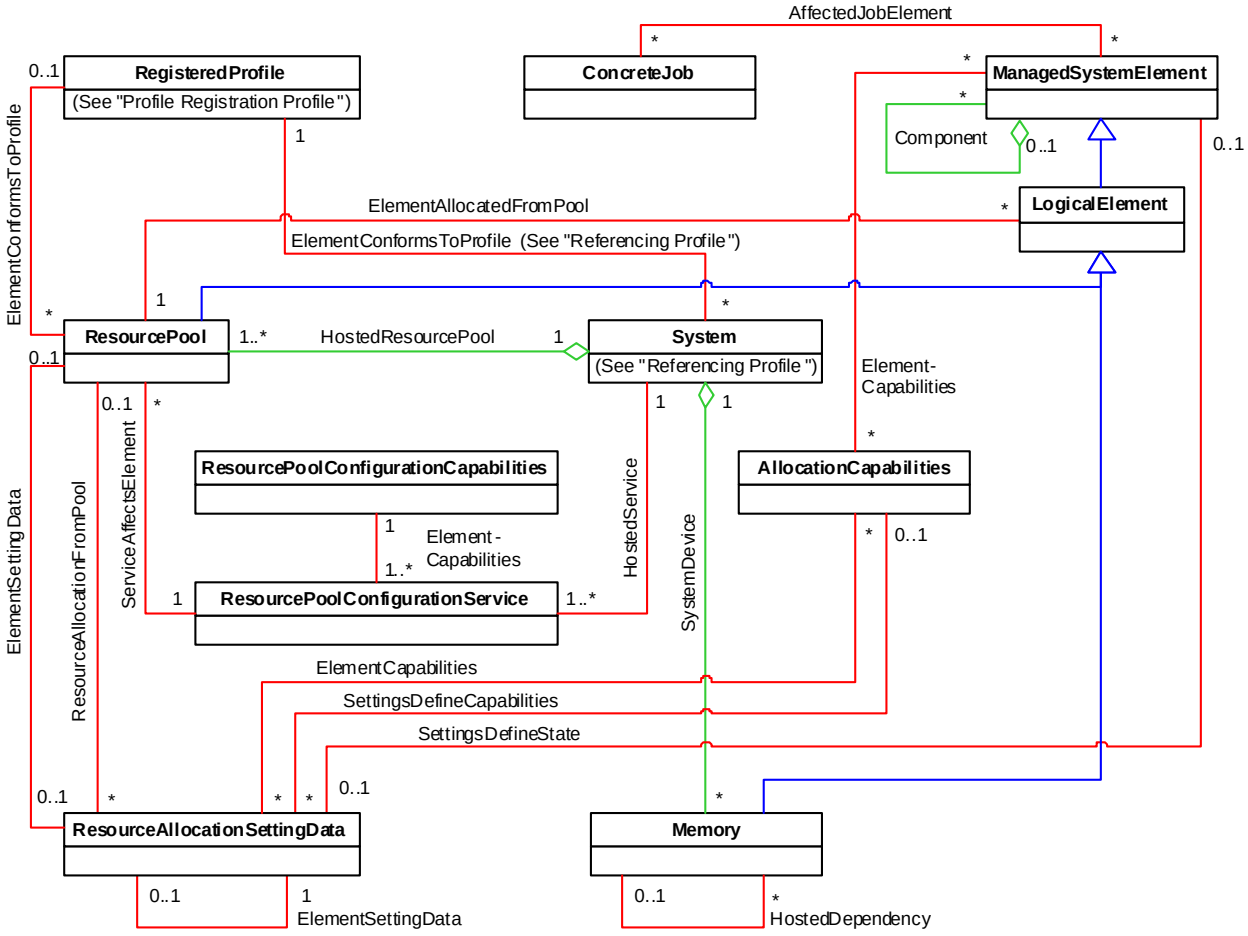

Figure 1 – Memory Resource Virtualization Profile: Profile Class Diagram

This profile specifies the use of the following classes and associations:

- The CIM_ResourcePool class modeling resource pools for memory resources. Host memory resources are allocated from their resource pool and used to create the memory resources for virtual systems
- The CIM_Component association modeling the following relationships:
  - the relationship between memory resource pools and memory extents as components of the resource pools

350 – the relationship between one aggregated memory extent and one or more aggregating
351 memory extents

352 • The CIM_ElementAllocatedFromPool association modeling hierarchies of memory resource
353 pools and modeling the relationship of resource pools and the virtual memory allocated from
354 those

355 • The CIM_HostedResourcePool association modeling the hosting dependency between a
356 memory resource pool and its host system. A host system supports at least one resource pool

357 • The CIM_Memory class modeling the following aspects of memory:

358 – memory as a device in the scope of a system, as modeled by the CIM_SystemDevice
359 association

360 – memory as a result of a memory resource allocation from a resource pool, as modeled by
361 the CIM_ElementAllocatedFromPool association

362 – memory as a component within memory resource pools, as modeled through the
363 CIM_Component association

364 • The CIM_ResourceAllocationSettingData class representing memory resource allocations or
365 memory resource allocation requests

366 • The CIM_AllocationCapabilities class and the CIM_ElementCapabilities association modeling

367 – the memory resource allocation capabilities of host systems

368 – the memory resource allocation capabilities of memory resource pools

369 – the mutability of existing memory allocations

370 • The CIM_SettingsDefineCapabilities association modeling the relation between memory
371 allocation capabilities and the settings that define these capabilities

372 • The CIM_ResourcePoolConfigurationService modeling configuration services for memory
373 resource pools and the CIM_ResourcePoolConfigurationCapabilities class modeling their
374 capabilities

375 • The CIM_ConcreteJob class and the CIM_AffectedJobElement association modeling
376 asynchronous management tasks initiated through memory resource pool configuration
377 services

378 In general, any mention of a class in this document means the class itself or its subclasses. For example,
379 a statement such as “an instance of the CIM_StorageExtent class” implies an instance of the
380 CIM_StorageExtent class or a subclass of the CIM_StorageExtent class.

381 6.3 Memory Resource Pool

382 This profile applies the concept of resource pools defined in DSP1041 to the memory resource type. The
383 *Memory Resource Virtualization Profile* uses the memory resource pool as the focal point for memory
384 allocations. Virtual systems receive memory allocations from memory resource pools based on memory
385 resource allocation requests. A memory resource pool is an aggregation of host memory available for
386 allocation in support of virtual memory.

387 Two types of memory resource pools are defined: primordial and concrete.

388 6.3.1 Primordial Memory Resource Pool

389 A primordial memory resource pool aggregates memory capacity; it represents a subset of the
390 manageable memory capacity of a host system.

6.3.2 Concrete Memory Resource Pool

A concrete memory resource pool subdivides the memory capacity of its parent resource pool. The amount of memory allocated to a concrete resource pool is less than or at most equal to the capacity of the parent pool, but may use all of the capacity of the parent pool.

6.3.3 Hierarchies of Memory Resource Pools

This profile applies the concept of resource pool hierarchies defined in DSP1041 to the memory resource type; see the “Hierarchies of Resource Pools” subclause in DSP1041.

Figure 2 shows an example of the CIM representation of a memory resource pool hierarchy in which two host memory extents are aggregated into a primordial memory resource pool. The primordial memory resource pool is subdivided into three concrete memory resource pools, with each concrete resource pool allocated 1 GB of memory. The assigned weights differ for each concrete memory resource pool, indicating different qualities of service for memory extents that are allocated from these pools.

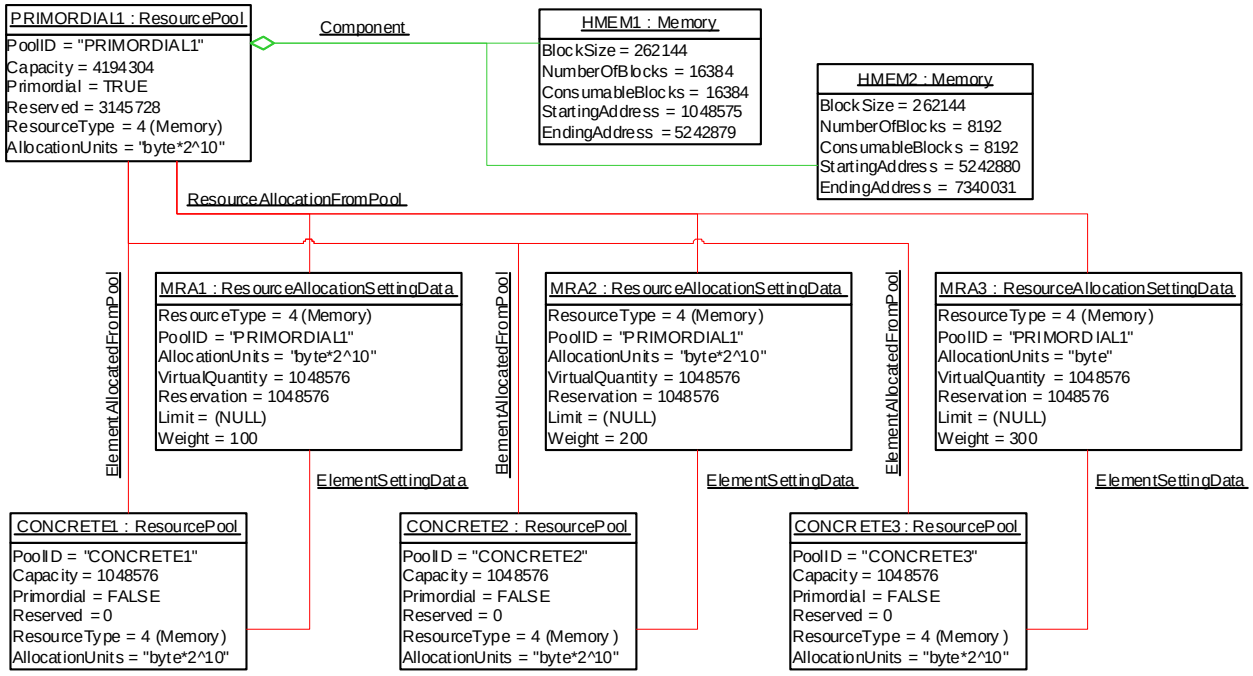

Figure 2 – Instance Diagram: Concept of Memory Resource Pool Hierarchies

6.3.4 Memory Resource Pool Management

This profile applies the concept of resource pool management defined in DSP1041 to the memory resource type; see the “Pool and Resource Management” subclause in DSP1041.

6.4 Memory Resource Allocation

This profile applies the concept of device resource allocation defined in DSP1041 to the memory resource type; see the “Device Resource Allocation” subclause in DSP1041.

6.4.1 General

Figure 3 shows a typical situation for memory resource allocation in the context of a virtual system.

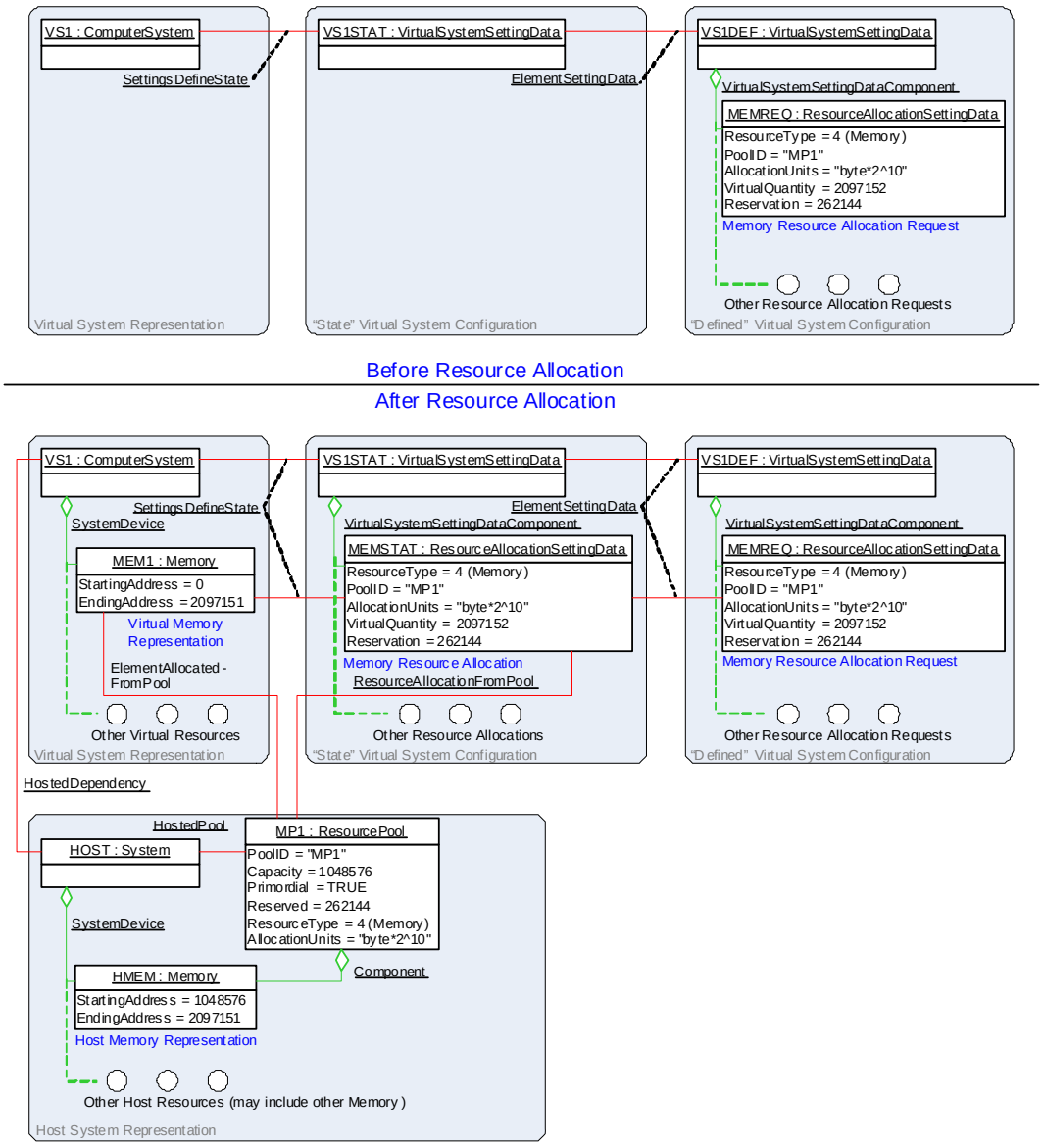

Figure 3 – Instance Diagram: Concept of Memory Resource Allocation

6.4.2 Memory Resource Allocation Request

The memory requirements of a virtual system are defined as part of the “Defined” virtual system configuration. The “Defined” virtual system configuration contains memory resource allocation requests represented as instances of the CIM_ResourceAllocationSettingData class.

An example of the CIM representation of a memory resource allocation request is shown in the upper right part of Figure 3.

6.4.3 Memory Resource Allocation

As a virtual system is activated (or instantiated), memory needs to be allocated as requested by memory resource allocation requests in the virtual system definition. Memory resource allocations are represented as instances of the CIM_ResourceAllocationSettingData class in the “State” virtual system configuration.

An example of the CIM representation of a memory resource allocation is shown in the center part of Figure 3.

6.4.4 Virtual Memory

Virtual memory is the instantiation of allocated host memory that is exposed to a virtual system through a logical memory device; it is the result of the memory resource allocation based on a memory resource allocation request. Virtual memory may be virtualized using techniques like paging and dynamic address translation, but may also be host memory that is directly passed through to the virtual system. Virtual memory is represented by an instance of the CIM_Memory class as part of the virtual system representation.

NOTE The definition of the term “virtual memory” is specialized from the term “virtual resource” defined in DSP1041 and deviates from common computer industry parlance.

An example of the CIM representation of virtual memory as the result of a memory resource allocation is shown on the left side in the central part of Figure 3.

6.4.5 Memory Virtualization

The amount of host memory reserved may be less than the amount of virtual memory available to the virtual system. This indicates that memory is virtualized by facilities of the host system, such that the amount of virtual memory usable by the virtual system is larger than the amount of real memory required for its support.

An example of the CIM representation of memory virtualization is shown in the center part of Figure 3 where the amount of virtual memory is significantly larger than the amount of allocated host memory.

6.4.6 Memory Composition

Memory may be composed of other memory. For example, a virtualization platform may support the allocation of memory from more than one resource pool, resulting in several virtual memory extents that then are composed into one aggregating memory extent. This situation is shown in Figure 4. It is similar to the situation shown in Figure 3, but in Figure 4 the virtual memory is composed from two memory extents that are allocated from two different memory resource pools. The two memory extents are composed into a memory composition that is a logical device of the virtual system.

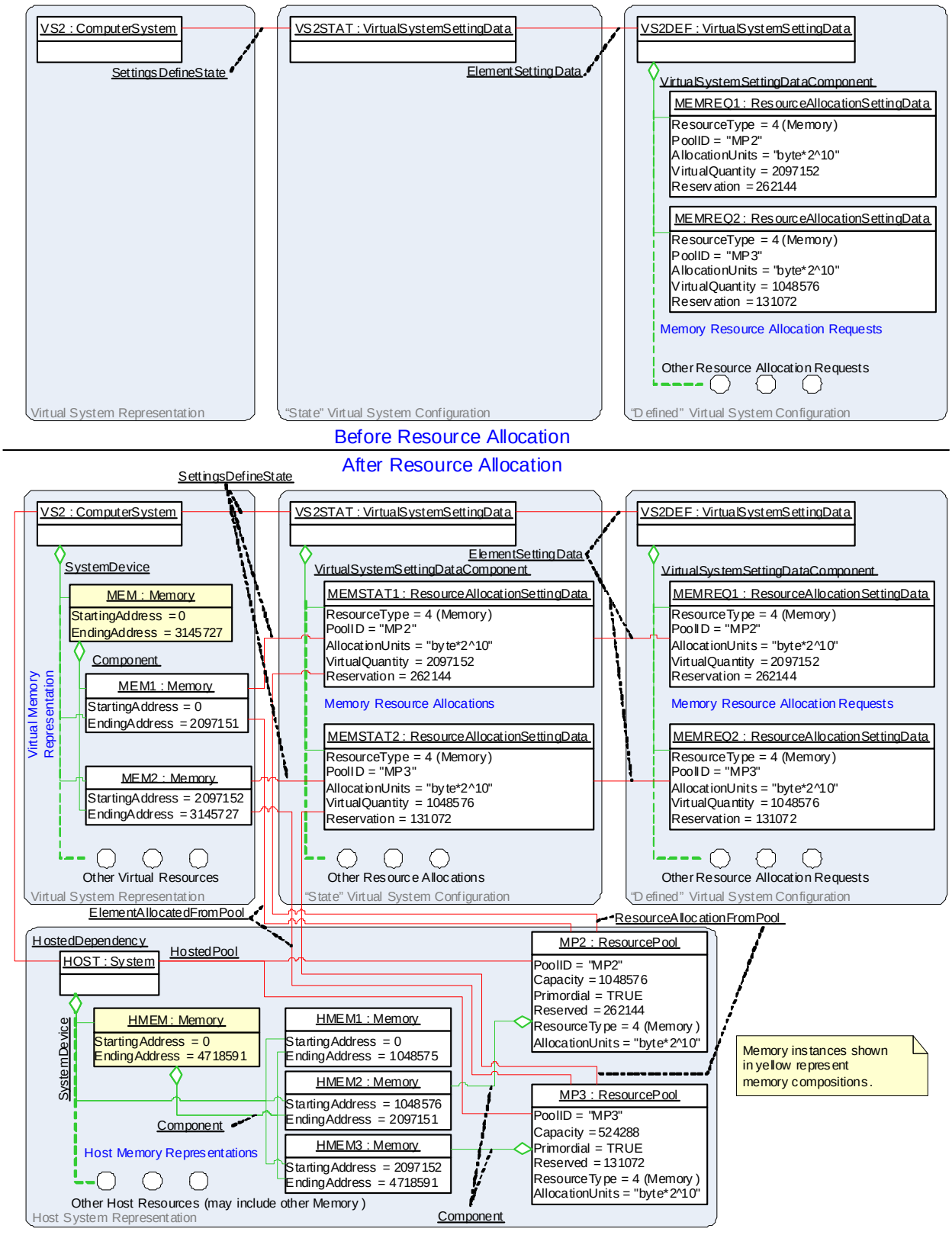

Figure 4 – Instance Diagram: Memory Composition

6.4.7 Dedicated Host Memory

Dedicated host memory is memory owned by the host system that is exclusively reserved for support of the virtual memory of a particular virtual system.

6.4.8 Host Memory Consumption

A memory resource allocation request references the memory resource pool to be used by specifying the value of the PoolID property. Host memory is allocated from the identified memory resource pool during memory resource allocation.

6.4.8.1 Statically Controlled Memory Consumption

With statically controlled memory consumption, a particular memory resource allocation request is supported only if the amount of host memory available in the addressed memory pool is at least as large as the amount requested. This approach is called *admission control*. Each successfully allocated memory resource reduces the amount of memory available in the pool, respectively. In the CIM representation of the memory resource pool, the amount of memory consumed from the pool is visible through the value of the Reserved property.

6.4.8.2 Dynamically Controlled Memory Consumption

With dynamically controlled memory consumption, the amount of host memory allocated in support of virtual memory is not a constant value but varies significantly over time, depending on factors like the memory access pattern of software executed within the virtual system or the memory consumption of other virtual machines.

Further, with dynamically controlled memory consumption, the amount of memory managed through a memory resource pool conceptually may be considered as unlimited, such that no admission control is performed at the time virtual memory is initially allocated (usually at virtual system activation time). Of course, the host system may experience memory shortages in later stages, such that ultimately the host system is no longer able to support the sum of virtual memory of all hosted virtual systems.

7 Implementation

This clause details the requirements related to classes and their properties for implementations of this profile. The CIM Schema descriptions for any referenced element and its sub-elements apply.

The list of all methods covered by this profile is in clause 8. The list of all classes and their elements that are covered by this profile is provided in clause 10.

In references to CIM Schema properties that enumerate values, the numeric value is normative and the descriptive text following it in parentheses is informative. For example, in the statement “If an instance of the CIM_ResourcePoolConfigurationCapabilities class contains the value 4 (DeleteResourcePool is supported) in an element of the SynchronousMethodsSupported[ ] array property”, the value “4” is normative text and “(DeleteResourcePool is supported)” is informative text.

7.1 Allocation Units

All properties that describe storage extents of memory shall be measured in the allocation unit “kilobyte”. This requirement applies to all the following classes and properties:

- CIM_Memory
  - StartingAddress property: Value shall be in units of kilobyte
  - EndingAddress property: Value shall be in units of kilobyte

NOTE: The properties NumberOfBlocks and ConsumableBlocks in the CIM_StorageExtent class that is base class of the CIM_Memory class allows specifying the memory size in units of blocks, with the blocksize specified through the value of the BlockSize property.

- CIM_ResourceAllocationSettingData
  - AllocationUnits property: Value shall be “byte*2^10” (equals kilobyte)
  - Reservation property: Value shall be in units of kilobyte
  - Limit property: Value shall be in units of kilobyte
- CIM_ResourcePool
  - AllocationUnits property: Value shall be “byte*2^10” (equals kilobyte)
  - Capacity property: Value shall be in units of kilobyte
  - Reserved property: Value shall be in units of kilobyte

7.2 Resource Allocation Profile

DSP1041 should be used to model host memory.

DSP1041 shall be used to model

- memory resource pool
- memory resource allocation
- memory resource allocation request
- virtual memory

7.2.1 Host Memory

The support of the representation of host memory is optional.

NOTE The support for the representation of host memory is mandatory if the *System Memory Profile* (DSP1026) is supported; see 7.4.

If the representation of host memory supported, all of the following rules apply:

1) Host memory shall be represented by one or more instances of the CIM_Memory class that are associated with the instance of the CIM_System class that represents the host system through an instance of the CIM_SystemDevice association.
2) If the host memory is composed from more than one extent, host memory shall be represented as a memory composition as follows:
   - Each composing memory extent shall be represented by an instance of the CIM_Memory class as required by rule 1).
   - Total memory shall be represented by an instance of the CIM_Memory class as required by rule 1). In that instance the value of the StartingAddress property shall be 0, and the value of the EndingAddress property shall be the highest ending address from any of the composing memory extents. The range spanned by subtracting the value of the EndingAddress property from that of the StartingAddress property shall be reflected by the values of the BlockSize, NumberOfBlocks and ConsumableBlocks properties, respectively.
   - Each instance if the CIM_Memory class representing a composing memory extent shall be associated with the instance of the CIM_Memory class representing total memory through an instance of the CIM_Component association.

NOTE: In a memory composition, total memory may span memory gaps that are not covered by a composing memory extent.

7.2.2 Memory Resource Pool

This subclause specifies implementation requirements for the representation of memory resource pools.

7.2.2.1 CIM_ResourcePool.Primordial Property

The value of the Primordial property shall be set to TRUE for any instance of the CIM_ResourcePool class that represents a primordial memory resource pool. For other instances of the CIM_ResourcePool class that represent memory resource pools, the value of the Primordial property shall be set to FALSE.

7.2.2.2 CIM_ResourcePool.PoolID Property

The value of the PoolID property shall be set such that it enables unique identification of the instance of the CIM_ResourcePool class within the scoping host system.

7.2.2.3 Aggregation of Host Resources

The support of the representation of the aggregation of host memory into a primordial memory resource pool is optional.

If the representation of the aggregation of host memory into primordial memory resource pools is supported, at least one instance of the CIM_Component association (see 10.5) references the instance of the CIM_ResourcePool class that represents the pool.

7.2.2.4 CIM_ResourcePool.Reserved Property

The value of the Reserved property shall denote the amount of host memory that is actually reserved from the resource pool, in units of kilobyte.

7.2.2.5 CIM_ResourcePool.Capacity Property

The support of the Capacity property is conditional.

Conditional Requirement: The Capacity property shall be supported if the aggregation of host resources is supported (see 7.2.2.3); otherwise, support of the Capacity property is optional.

If the Capacity property is supported, its value shall reflect the maximum amount of memory that can be allocated from the resource pool in units of kilobyte. If the instance of the CIM_ResourcePool class represents a memory resource pool with unlimited capacity, the value of the Capacity property shall be set to the largest value supported by the uint64 datatype.

7.2.2.6 Memory Resource Pool Hierarchies

The support of representing memory resource pool hierarchies is optional.

If the representation of memory resource pool hierarchies is supported, any concrete memory resource pool shall be represented through an instance of the CIM_ResourcePool class, where all of the following conditions shall be met:

- The value of the Primordial property shall be FALSE.
- The instance shall be associated through an instance of CIM_ElementAllocatedFromPool association to the instance of the CIM_ResourcePool class that represents its parent memory resource pool.
- The instance shall be associated through an instance of the CIM_ElementSettingData association to the instance of the CIM_ResourceAllocationSettingData class that represents the amount of memory allocated from the parent resource pool.

7.2.2.7 Default Memory Resource Pool

The support of designating a default memory resource pool is optional.

If the designation of a default memory resource pool is supported, all of the following conditions apply:

- The default memory resource pool shall be represented by an instance of the CIM_ResourcePool class; see 10.18.
- That instance shall be associated to the instance of the CIM_AllocationCapabilities class that represents the pools default allocation capabilities as specified in 7.3.1.5.
- The same instance of the CIM_AllocationCapabilities class shall also represent the systems default allocation capabilities as specified in 7.3.1.2.

7.2.2.8 Memory Resource Pool Management

The support of memory resource pool management is optional.

7.2.2.8.1 Indication of Support

If memory resource pool management is supported, the instance of the CIM_ResourcePoolManagementCapabilities class that is associated through an instance of the CIM_ElementCapabilities association to the instance of the CIM_ResourcePoolManagementService that represents a memory resource pool configuration service shall represent the capabilities of the resource pool configuration service as specified in this subclause.

Memory resource pool management is supported (with one or more methods) if the SynchronousMethodsSupported[ ] array property, the AsynchronousMethodsSupported[ ] array property, or both have a non-NULL value and contain at least one element.

If memory resource pool management is not supported, the value of both the SynchronousMethodsSupported[ ] and the AsynchronousMethodsSupported[ ] array properties shall be NULL or an empty array in the instance of the CIM_ResourcePoolManagementCapabilities class that represents the capabilities of a resource pool configuration service.

7.2.2.8.2 ValueMap Qualifier Method Designators

Qualifier values defined by the ValueMap qualifiers of the SynchronousMethodsSupported[ ] and AsynchronousMethodsSupported[ ] array properties in the CIM_ResourcePoolConfigurationCapabilities class shall designate methods as follows:

- The value 3 (CreateChildResourcePool is supported) designates the CreateChildResourcePool( ) method.
- The value 4 (DeleteResourcePool is supported) designates the DeleteResourcePool( ) method.
- The value 5 (AddResourcesToResourcePool is supported) designates the AddResourcesToResourcePool( ) method.
- The value 6 (RemoveResourcesFromResourcePool) designates the RemoveResourcesFromResourcePool( ) method.

7.2.2.8.3 Implementation Requirements

Elements in the value sets of the SynchronousMethodsSupported[ ] and AsynchronousMethodsSupported[ ] array properties in the CIM_ResourcePoolConfigurationCapabilities class shall be specified according to the following rules:

- A particular ValueMap qualifier value shall appear as an element in the value set of at most one array property.

- If a particular ValueMap qualifier value does not appear in the value set of either array property, the corresponding method is not supported.
- If a particular qualifier value is used as element in the value set of the SynchronousMethodsSupported[ ] array property, the corresponding method shall be supported with synchronous behavior only.
- If a particular qualifier value is used as element in the value set of the AsynchronousMethodsSupported[ ] array property, the corresponding method shall be supported with synchronous behavior, asynchronous behavior, or both.

A method implementation shall apply following rules:

- A supported method (with synchronous or asynchronous behavior) shall not return 1 (Not Supported).
- A method supported with synchronous behavior only shall not return 4096 (Method Parameters Checked – Job Started).

7.2.2.8.4 Availability of Support of Asynchronous Operations

This subclause specifies the CIM elements that indicate the availability of the support for asynchronous operations for memory resource pool management.

Asynchronous operations for memory resource pool management are supported (with one or more methods) if the AsynchronousMethodsSupported[ ] array property has a non-NULL value and contains at least one element.

7.2.3 Memory Resource Allocation

NOTE: DSP1041 specifies two alternatives for modeling resource allocation: *simple resource allocation* and *virtual resource allocation*.

Implementations conforming to the *Memory Resource Virtualization Profile* shall implement virtual resource allocation as defined in the “Modeling Virtual Resource Allocation (Optional)” clause of DSP1041.

7.2.4 CIM_ResourceAllocationSettingData

This subclause specifies the use of the CIM_ResourceAllocationSettingData class.

7.2.4.1 General

Instances of the CIM_ResourceAllocationSettingData class

- shall be used to represent memory resource allocation requests (MRQ)
- shall be used to represent memory resource allocations (MRA)
- shall be used to represent settings that define the capabilities of systems and of memory resource pools (MCA)
- may be used to represent settings that define the mutability of memory resource allocations (MMS)

The specifications in this subclause generally define constraints for property values used in these representations. Constraints that apply to only a subset of these representations are prefixed with the respective acronyms and followed by the word “Only” and a colon.

7.2.4.2 CIM_ResourceAllocationSettingData.PoolID Property

The value of the PoolID property shall designate the memory resource pool. A NULL value shall indicate the use of the host system's default memory resource pool.

7.2.4.3 CIM_ResourceAllocationSettingData.ConsumerVisibility Property

The value of the ConsumerVisibility property shall denote whether host memory is directly passed through to the virtual system or whether memory is virtualized. Values shall be assigned as follows:

- A value of 2 (Passed-Through) shall denote that host memory is passed through.
- A value of 3 (Virtualized) shall denote that memory is virtualized.
- MRQ Only: A value of 0 (Unknown) shall be used if the represented memory resource allocation request does not predefine which type of memory shall be allocated.

Other values shall not be used.

7.2.4.4 CIM_ResourceAllocationSettingData.HostResource[ ] Array Property

Support of the HostResource[ ] array property is conditional if representing memory resource allocations, and optional otherwise.

MRA Only: Conditional Requirement: The HostResource[ ] array property shall be supported if the value of the MappingBehavior property is 2 (Dedicated).

Otherwise, the HostResource[ ] array property may be supported.

If HostResource[ ] array property is supported, the following rules apply:

- If the value of the ConsumerVisibility property (see 7.2.4.3) is 2 (Passed-Through), the value of the HostResource[ ] array property should designate the host memory resource that is passed through to a virtual system. A value of NULL or an empty array should indicate that the passed-through host memory resource is not represented by an instance of the CIM_Memory class.
- If the value of the ConsumerVisibility property is 3 (Virtualized), the value of the HostResource[ ] array property shall be NULL or shall be an empty array.
- MRA Only: The value of the HostResource[ ] array property depends on the value of the MappingBehavior property as follows:
  - If the value of the MappingBehavior property is 2 (Dedicated), elements in the value of the HostResource[ ] array property shall reference instances of the CIM_Memory class that represent host memory extents that are exclusively dedicated to the virtual system.
  - If the value of the MappingBehavior property is 3 (Hard Affinity) or 4 (Soft Affinity), elements in the value of the HostResource[ ] array property shall reference instances of the CIM_Memory class that represent host memory extents that provide the allocation of the virtual memory.
- MRQ Only: The value of the HostResource[ ] array property depends on the value of the MappingBehavior property as follows:
  - If the value of the MappingBehavior property is 2 (Dedicated), elements in the value of the HostResource[ ] array property shall reference instances of the CIM_Memory class that represent host memory extents that are required with dedicated access by the virtual system.
  - If the value of the MappingBehavior property is 3 (Soft Affinity), elements in the value of the HostResource[ ] array property shall reference instances of the CIM_Memory class that

represent host memory extents that are preferred for the allocation of the virtual systems memory.

– If the value of the MappingBehavior property is 4 (Hard Affinity), elements in the value of the HostResource[ ] array property shall reference instances of the CIM_Memory class that represent host memory extents that are required for the allocation of the virtual systems memory.

If the HostResource[ ] array property is not supported, the value of the HostResource[ ] array property shall be NULL. This indicates that host memory extents that are exclusively dedicated to the virtual system or that provide the allocation of the virtual memory are not defined.

7.2.4.5 CIM_ResourceAllocationSettingData.VirtualQuantity Property

The value of the VirtualQuantity property shall denote the amount of virtual memory available to a virtual system in units of kilobyte.

7.2.4.6 CIM_ResourceAllocationSettingData.Reservation Property

Support of the Reservation property is optional.

If the Reservation property is supported, the value of the Reservation property shall denote the amount of host memory reserved for the exclusive use of a virtual system in units of kilobyte.

If the Reservation property is not supported, it shall have a value of NULL. This indicates that an amount of host memory reserved for the exclusive use of the virtual system is not defined.

7.2.4.7 CIM_ResourceAllocationSettingData.Limit Property

Support of the Limit property is optional.

If the Limit property is supported, the value of the Limit property shall denote the maximum amount of host memory available to a virtual system in units of kilobyte.

If the Limit property is not supported, it shall have a value of NULL. This indicates that a maximum amount of host memory available to the virtual system is not defined.

7.2.4.8 CIM_ResourceAllocationSettingData.Weight Property

Support of the Weight property is optional.

If the Weight property is supported, its value shall denote the relative priority of a memory resource allocation in relation to other memory resource allocations.

If the Weight property is not supported, it shall have a value of NULL. This indicates that a relative priority of the memory resource allocation in relations to other memory resource allocations is not defined.

7.2.4.9 CIM_ResourceAllocationSettingData.Parent Property

The implementation of the Parent property is optional.

If the Parent property is supported, the value of the Parent property shall denote the parent entity of the memory resource allocation.

If the Parent property is not supported, is shall have a value of NULL. This indicates that a parent entity of the memory resource allocation is not defined.

NOTE: For example, the value of the Parent property may refer to the name of an address space that resides in the host system.

7.2.4.10 CIM_ResourceAllocationSettingData.Connection[ ] Array Property

The implementation of the Connection[ ] array property is optional.

If Connection[ ] array property is supported, its value shall contain elements that identify entities connected to the memory resource allocation.

If the Connection[ ] array property is not supported, it shall have a value of NULL. This indicates that entities connected to the memory resource allocation are not defined.

NOTE: For example, elements of the value of the Connection[ ] array property may refer to the name of shared memory segments that are mapped to the allocated virtual memory.

7.2.4.11 CIM_ResourceAllocationSettingData.MappingBehavior Property

The implementation of the MappingBehavior property is optional.

If the MappingBehavior property is supported, its value shall denote how host resources referenced by elements in the value of HostResource[ ] array property relate to the memory resource allocation. The following rules apply:

- MRA Only:
  - A value of 2 (Dedicated) shall indicate that the represented memory resource allocation is provided by host memory resources as referenced by the value of the HostResource[ ] array property that are exclusively dedicated to the virtual system.
  - A value of 3 (Soft Affinity) or 4 (Hard Affinity) shall indicate that the represented memory resource allocation is provided using host memory resource as referenced by the value of the HostResource[ ] array property.
  - Other values shall not be used.
- MRQ Only:
  - A value of 0 (Unknown) shall indicate that the memory resource allocation request does not require specific host resources.
  - A value of 2 (Dedicated) shall indicate that the memory resource allocation request shall be provided by exclusively dedicated host memory resources as specified through the value of the HostResource[ ] array property.
  - A value of 3 (Soft Affinity) shall indicate that the memory resource allocation request shall preferably be provided by host memory resources as specified through the value of the HostResource[ ] array property, but that other resources may be used if the requested resources are not available.
  - A value of 4 (Hard Affinity) shall indicate that the memory resource allocation request shall preferably be provided by host memory resources as specified through the value of the HostResource[ ] array property and that other resources shall not be used if the requested resources are not available.
  - Other values shall not be used.

If the MappingBehavior property is not supported, it shall have a value of NULL. This indicates that a further qualification of the value of the HostResource[ ] array property through the value of the MappingBehavior property is not defined.

7.2.5 Virtual Memory

For the representation of virtual memory all of the following rules apply:

1) Virtual memory shall be represented by one or more instances of the CIM_Memory class that are associated with the instance of the CIM_ComputerSystem class that represents the virtual system through an instance of the CIM_SystemDevice association.

2) If the virtual memory is composed from more than one extent, virtual memory shall be represented as a memory composition as follows:

   – Each composing memory extent shall be represented by an instance of the CIM_Memory class as required by rule 1).

   – Total memory shall be represented by an instance of the CIM_Memory class as required by rule 1). In that instance the value of the StartingAddress property shall be 0, and the value of the EndingAddress property shall be the highest ending address from any of the composing memory extents. The range defined by subtracting the value of the EndingAddress property from that of the StartingAddress property shall be reflected by the values of the BlockSize, NumberOfBlocks and ConsumableBlocks properties, respectively.

   – Each instance of the CIM_Memory class representing a composing memory extent shall be associated with the instance of the CIM_Memory class representing total memory through an instance of the CIM_Component association.

3) If a memory extent is directly based on a memory resource allocation, the instance of the CIM_Memory class representing that extent shall be associated to all of the following instances:

   – the instance of the CIM_ResourceAllocationSettingData that represents memory resource allocation through an instance of the CIM_SettingsDefineState association

   – the instance of the CIM_ResourcePool that represents the memory resource pool providing the resource allocation through an instance of the CIM_ElementAllocatedFromPool association

NOTE 1: In a memory composition, total memory may span memory gaps that are not covered by a composing memory extent. Discontiguous memory as seen in the memory address space presented to the virtual system is not supported.

NOTE 2: In a memory composition, total memory is never the direct result of a memory allocation; instead, each composing memory extent is the direct result of a memory allocation.

Additional constraints apply if DSP1026 is implemented; see 7.4.

7.3 Allocation Capabilities Profile

DSP1043 shall be used to model the following aspects:

- the memory resource allocation capabilities of host systems
- the memory resource allocation capabilities of memory resource pools
- the mutability of memory resource allocations or memory resource allocation requests

7.3.1 Memory Resource Allocation Capabilities

The memory resource allocation capabilities of host systems and of memory resource pools shall be represented by instances of the CIM_AllocationCapabilities class.

7.3.1.1 CIM_AllocationCapabilities Class (Capabilities)

This subclause specifies the use of the CIM_AllocationCapabilities class for the representation of the memory resource allocation capabilities of host systems and of memory resource pools.

7.3.1.1.1 Relationship of System and Resource Pool Capabilities

The memory allocation capabilities of a host system shall be a superset of the memory allocation capabilities of all memory resource pools that are hosted by the host system.

7.3.1.1.2 CIM_AllocationCapabilities.RequestedTypesSupported Property

The value of the RequestedTypesSupported property shall indicate whether the host system or the resource pool support memory resource allocation requests for particular host resources, as follows:

- A value of 2 (Specific) shall indicate support of specific requests that only refer to a specific host memory resource.
- A value of 3 (General) shall indicate support of generic memory requests that do not refer to a particular host memory resource.
- A value of 4 (Both) shall indicate support of both specific and generic memory requests.

Other values shall not be used.

7.3.1.1.3 CIM_AllocationCapabilities.SharingMode Property

The value of the SharingMode property shall indicate whether the host system or the resource pool support exclusive access or shared use of managed memory resources, as follows:

- A value of 2 (Dedicated) shall indicate support of memory resources with exclusive access. This value shall be used if host memory is allocated directly to a virtual system for exclusive use.
- A value of 3 (Shared) shall indicate support of memory resources with shared access. This value shall be used if host memory is not allocated directly to a virtual system.

Other values shall not be used.

NOTE: More than one instance of the CIM_AllocationCapabilities class may be associated with a host system or resource pool. As a result, support of both shared and exclusive access to memory resources may be modeled, and one of these sharing modes may be designated the default sharing mode; see 7.3.1.3.

7.3.1.2 Host System Memory Resource Allocation Capabilities

Each instance of the CIM_System class that represents a host system shall be associated through the CIM_ElementCapabilities association with one or more instances of the CIM_AllocationCapabilities class that represent the host system's memory resource allocation capabilities. One of these instances shall represent the default memory resource allocation capabilities as specified in 7.3.1.3.

7.3.1.3 Default Host System Memory Resource Allocation Capabilities

Exactly one instance of the CIM_AllocationCapabilities class shall exist that describes the default memory resource allocation capabilities of a host system. That instance shall be associated with the instance of the CIM_System class that represents the host system through an instance of the CIM_ElementCapabilities association where the value of the Characteristics[ ] array property shall contain exactly one element, and that element shall have a value of 2 (Default).

7.3.1.4 Memory Resource Pool Memory Resource Allocation Capabilities

Each instance of the CIM_ResourcePool class that represents a memory resource pool shall be associated through the CIM_ElementCapabilities association with one or more instances of the CIM_AllocationCapabilities class that represent the memory resource pool's memory resource allocation capabilities. One of these instances shall represent the default memory resource allocation capabilities as specified in 7.3.1.5.

7.3.1.5 Default Memory Resource Pool Memory Resource Allocation Capabilities

Exactly one instance of the CIM_AllocationCapabilities class shall exist that describes the default memory resource allocation capabilities of a memory resource pool. That instance shall be associated with the instance of the CIM_ResourcePool class that represents the memory resource pool through an instance of the CIM_ElementCapabilities where the value of the Characteristics array property shall contain exactly one element, and that element shall have a value of 2 (Default).

7.3.2 Memory Resource Allocation Mutability

The support for the representation of the mutability of memory resource allocation requests and memory resource allocations is optional.

7.3.2.1 Indication of Support

If the representation of the mutability of memory resource allocation requests and memory resource allocations is supported, in both cases it shall be represented by instances of the CIM_AllocationCapabilities class that are associated with the instance of the CIM_ResourceAllocationSettingData class that represents the memory resource allocation request or memory resource allocation through instances of the CIM_ElementCapabilities association.

If the representation of the mutability of a memory resource allocation or a memory resource allocation request is not supported, the instance of the CIM_ResourceAllocationSettingData class that represents the memory resource allocation request or the memory resource allocation shall not be associated with an instance of the CIM_AllocationCapabilities class through an instance of the CIM_ElementCapabilities association.

7.3.2.2 CIM_AllocationCapabilities Class (Mutability)

This subclause specifies the use of the CIM_AllocationCapabilities class for the representation of the mutability of a memory resource allocation request or a memory resource allocation.

7.3.2.2.1 CIM_AllocationCapabilities.RequestedTypesSupported Property

The value of the RequestedTypesSupported property shall indicate whether the type of the memory resource allocation can be changed, as follows:

- A value of 2 (Specific) shall indicate support of change requests that refer to a specific host memory resource.
- A value of 3 (General) shall indicate support of change requests that do not refer to a particular host memory resource.
- A value of 4 (Both) shall indicate support of either type of change request.

Other values shall not be used.

7.3.2.2.2 CIM_AllocationCapabilities.SharingMode Property

The value of the SharingMode property shall indicate whether the sharing mode of the memory resource allocation can be changed, as follows:

- A value of NULL shall indicate that the sharing mode of the memory allocation can not be changed.
- A value of 2 (Dedicated) shall indicate support of requests to change the sharing mode to exclusive access.
- A value of 3 (Shared) shall indicate support of requests to change the sharing mode to shared access.

893 Other values shall not be used.

894 **7.3.2.2.3 CIM_AllocationCapabilities.SupportedAddStates[ ] Array Property**

895 If the addition of memory to the described memory resource allocation or memory resource allocation
896 request is not supported, then If the value of the SupportedAddStates[ ] array property shall be NULL.

897 If the value set of the SupportedAddStates[ ] is empty, then the addition of memory to the described
898 memory resource allocation or memory resource allocation request shall be supported regardless of the
899 virtual system state of the scoping virtual system.

900 If values are provided as elements of the SupportedAddStates[ ] array property, these values shall
901 designate a set of potential virtual system states. The addition of memory to the described memory
902 resource allocation or memory resource allocation request shall be supported if the scoping virtual system
903 is in any of the designated virtual system states.

904 **7.3.2.2.4 CIM_AllocationCapabilities.SupportedRemoveStates[ ] Array Property**

905 If the removal of memory from the described memory resource allocation or memory resource allocation
906 request is not supported, then the value of the SupportedRemoveStates[ ] array property shall be NULL.

907 If the value set of the SupportedRemoveStates[ ] array property is empty, then the removal of memory
908 from the described memory resource allocation or memory resource allocation request shall be supported
909 regardless of the virtual system state of the scoping virtual system.

910 If values are provided as elements of the SupportedRemoveStates[ ] array property, these values shall
911 designate a set of potential virtual system states. The removal of memory from the described memory
912 resource allocation or memory resource allocation request shall be supported if the scoping virtual system
913 is in any of the designated virtual system states.

914 ## 7.4 System Memory Profile

915 The support of DSP1026 for representation of host system memory is optional.

916 The support of DSP1026 for representation of virtual system memory is optional. Additional constraints
917 defined in 7.2.5 apply.

918 NOTE: DSP1026 defines that there is exactly one instance of the CIM_Memory class associated to the instance of
919 the CIM_System class representing a system through an instance of the CIM_SystemDevice association. In context
920 of the *Memory Resource Virtualization Profile* it is expected that there are host systems as well as virtual systems
921 that have more than one memory extent assigned. A possible solution would be to use a memory composition where
922 the instance of the CIM_Memory class that represents the memory composition is assigned as central instance of
923 DSP1026.

924 # 8 Methods

925 This section details the requirements for supporting intrinsic operations and extrinsic methods for the CIM
926 elements defined by this profile.

927 ## 8.1 General

928 Support of intrinsic and extrinsic methods is specified by the “Methods” clauses in DSP1041 and
929 DSP1043. The only class added through the *Memory Resource Virtualization Profile* is the
930 CIM_RegisteredProfile class; see 10.15.

931 This section details the requirements for supporting intrinsic operations and extrinsic methods for the CIM
932 elements defined by this profile.

8.2 Profile conventions for operations

For each profile class (including associations), the implementation requirements for operations, including for those in the following default list, are specified in class-specific subclauses of this clause.

The default list of operations for all classes is:

GetInstance( )

EnumerateInstances( )

EnumerateInstanceNames( )

For classes that are referenced by an association, the default list also includes

Associators( )

AssociatorNames( )

References( )

ReferenceNames( )

The implementation requirements for intrinsic operations and extrinsic methods of classes listed in clause 10, but not addressed by a separate subclause of this clause are specified by the "Methods" clauses of respective base profiles, namely DMTF DSP1041 (*Resource Allocation Profile*) and DMTF DSP1043 (*Allocation Capabilities Profile*). These profiles are specialized by this profile, and in these cases this profile does not add method specifications beyond those defined in its base profiles.

8.3 CIM_RegisteredProfile

All operations in the default list in 8.2 are supported as described by DMTF DSP0200.

9 Use Cases (Informative)

The following use cases and object diagrams illustrate use of this profile. They are for informative purposes only and do not introduce behavioral requirements for implementations of the profile.

9.1 Object Diagram

Figure 5 depicts the CIM representation of a host system with one memory resource pool and one virtual system. Only information relevant in the context of memory resource virtualization is shown.

In Figure 5 the host system is represented by an instance of the CIM_System class tagged HOST. The host system owns two memory resources represented by instances of the CIM_Memory class that are tagged HOST_OWN and HOST_POOL. The first instance represents a memory extent in the lower part of the host memory that is used exclusively by the host because it is not assigned to a memory resource pool. The second instance represents a memory extent in the higher part of the host memory that is assigned to a memory resource pool.

The host system hosts a primordial memory resource pool represented by an instance of the CIM_ResourcePool class tagged MEM_POOL; in that instance the value of the PoolID property is “MEM_POOL”. The host memory resource that is represented by the instance of the CIM_Memory class tagged HOST_POOL is aggregated into the pool through an instance of the CIM_Component association.

The allocation capabilities of the host system and of the memory pool are represented by the same instance of the CIM_AllocationCapabilities class tagged CAP. Four instances of the CIM_ResourceAllocationSettingData class (tagged CAP_DEF, CAP_MIN, CAP_MAX, and CAP_INC) are associated with that instance through instances of the CIM_SettingsDefineCapabilities association. The

values of the ValueRange and ValueRole properties in the association instances designate the referenced instances of the CIM_ResourceAllocationSettingData class that represent the default, minimum, maximum, and increment for memory resource allocations that are supported by the system and the pool.

The host system hosts a virtual system that is represented by an instance of the CIM_ComputerSystem class tagged VS. The hosted relationship is shown through an instance of the CIM_HostedDependency association.

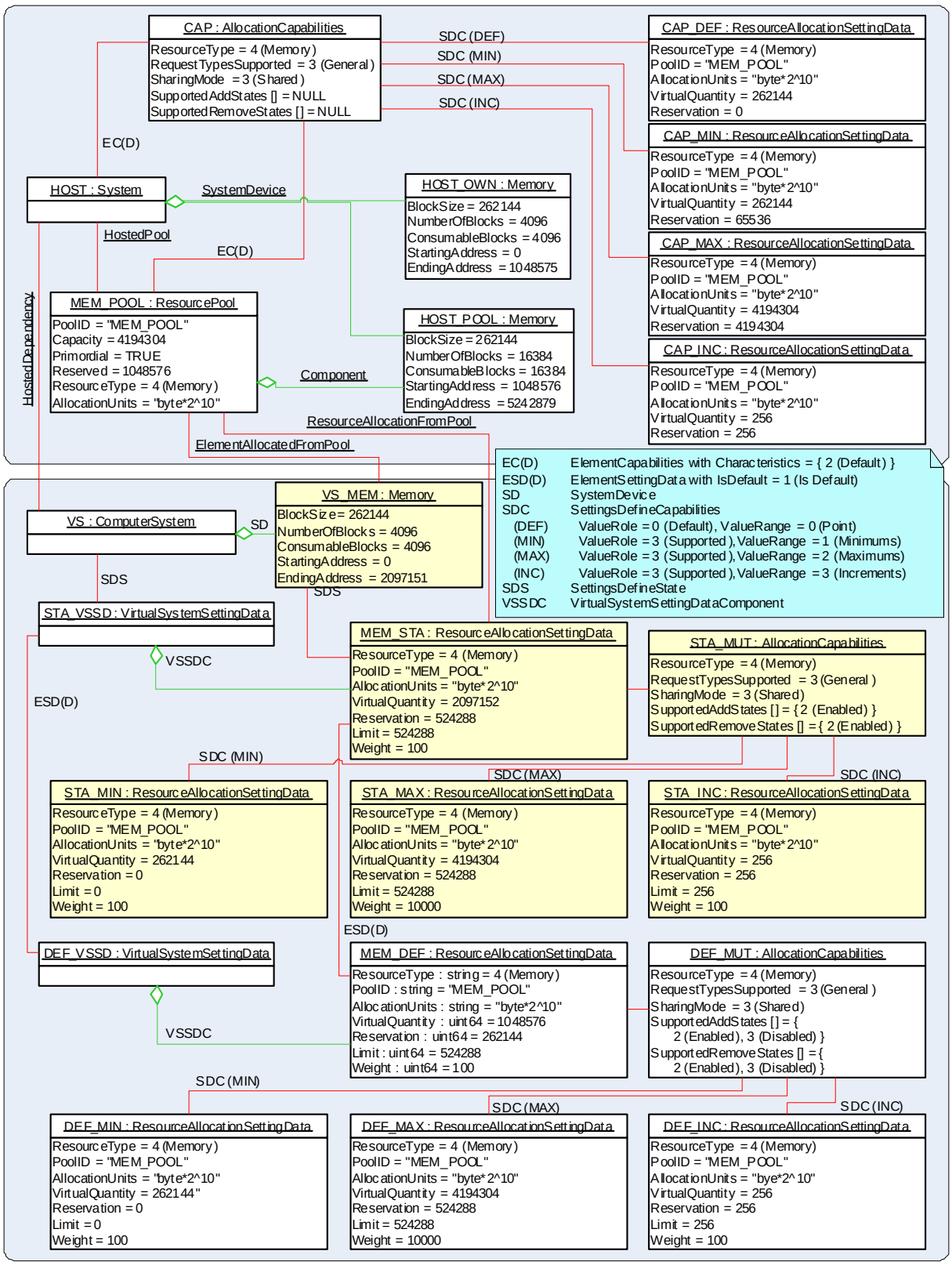

Figure 5 – Instance Diagram: Example CIM Representation of Memory Resource Virtualization

The head element of the “State” virtual system configuration is the instance of the CIM_VirtualSystemSettingData class tagged STA_VSSD; it is associated with the instance of the CIM_ComputerSystem class through an instance of the CIM_SettingsDefineState association. The “State” virtual system configuration contains an element of the CIM_ResourceAllocationSettingData class tagged MEM_STA that represents the memory resource allocation assigned to the virtual system. The virtual memory that results from the memory resource allocation is represented as part of the virtual system representation by the instance of the CIM_Memory class tagged VS_MEM.

NOTE: All instances in Figure 5 that are marked with light yellow color represent “State” entities that exist only as long as the virtual system is active (that is, in a state other than “Defined”). These instances do not exist while the virtual system is in the “Defined” state (that is, it is not instantiated).

The head element of the “Defined” virtual system configuration is the instance of the CIM_VirtualSystemSettingData class tagged DEF_VSSD; it is associated with the head element of the “State” virtual system configuration through an instance of the CIM_ElementSettingData association where the value of the IsDefault property is 1 (Is Default). The “Defined” virtual system configuration contains an element of the CIM_ResourceAllocationSettingData class tagged MEM_DEF that represents the memory resource allocation request that defines the memory requirements of the virtual system. That definition is used when the virtual system is activated and host memory is allocated to support the virtual system’s virtual memory.

The example in Figure 5 shows a situation in which the VirtualQuantity property in the instances of the CIM_ResourceAllocationSettingData class in the “State” and the “Defined” virtual system configuration has different values. This indicates that the memory size was dynamically modified (for example, by an operator command) sometime after the virtual system activation. This reflects a temporary situation that is retained until the virtual system is recycled, at which point memory resources are deallocated and then newly allocated based on the memory resource allocation request in the virtual system definition.

The mutability of both the memory resource allocation request in the “Defined” virtual system configuration and of the memory resource allocation in the “State” virtual system configuration is represented by instances of the CIM_ResourceAllocationSettingData class associated through respectively parameterized instances of the CIM_SettingsDefineCapabilities association.

Acceptable virtual system states for the addition and the removal of virtual memory are different for the memory resource allocation request and the memory resource allocation. The memory resource allocation can be modified only while the virtual system remains instantiated, as indicated by a value of 2 (Enabled) in the instance of the CIM_AllocationCapabilities class tagged STA_MUT.

9.2 Inspection

This set of use cases describes how to obtain various CIM instances that represent memory-related information of host and virtual systems.

9.2.1 Obtain the Memory Size of an Active Virtual System

Assumption: All of the following:

- The client knows a reference to the instance of the CIM_ComputerSystem class that represents the virtual system.
- The virtual system is in a virtual system state other than “Defined” (that is, is instantiated), which is indicated by a value other than 3 (Disabled) for the EnabledState property in the instance of the CIM_ComputerSystem class.

The sequence of activities is as follows:

1) The client resolves the CIM_SystemDevice association to find instances of the CIM_Memory class that represent the virtual memory, invoking the intrinsic Associators( ) CIM operation with parameter values set as follows:

– The value of the ObjectName parameter refers to the instance of the CIM_ComputerSystem class that represents the virtual system.

– The value of the AssocClass parameter is set to “CIM_SystemDevice”.

– The value of the ResultClass parameter is set to “CIM_Memory”.

The result of step 1) is a set of instances of the CIM_Memory class.

2) For each of the instances returned from step 1), the client inspects the values of the BlockSize and NumberOfBlocks properties, multiplies these values, and adds the results. The resulting sum is the amount of virtual memory available to the virtual system.

Result: The client knows the amount of virtual memory available to the virtual system.

In the example CIM representation shown in Figure 5, the client initially would know the instance of the CIM_ComputerSystem class tagged VS that represents the virtual system. From there, the client would follow the CIM_SystemDevice association to the instance of the CIM_Memory class tagged VS_MEM that represents the only virtual memory resource allocated to the virtual system. The client would multiply the value of the BlockSize property (524288) with the value of the ConsumableBlocks property (4096), yielding a result of 2147483648 bytes or 2097152 KB for the virtual system memory size.

9.2.2 Obtain the Memory Size of a Defined Virtual System

Assumption: All of the following:

- The client knows a reference to the instance of the CIM_ComputerSystem class that represents the virtual system.
- The virtual system is in the “Defined” virtual system state (that is, is not instantiated), which is indicated by a value of 3 (Disabled) for the EnabledState property in the instance of the CIM_ComputerSystem class.

The sequence of activities is as follows:

1) The client resolves the CIM_SettingsDefineState association to find the instance of the CIM_VirtualSystemSettingData class that is the head element of the “State” virtual system configuration, invoking the intrinsic AssociatorNames( ) CIM operation with parameter values set as follows:

   – The value of the ObjectName parameter refers to the instance of the CIM_ComputerSystem class that represents the virtual system.

   – The value of the AssocClass parameter is set to “CIM_SettingsDefineState”.

   – The value of the ResultClass parameter is set to “CIM_VirtualSystemSettingData”.

   The result of step 1) is a reference to an instance of the CIM_VirtualSystemSettingData class.

2) The client obtains instances of the CIM_ElementSettingData association that reference the result instance from step 1) to find the instance of the CIM_VirtualSystemSettingData class that is the head element of the “Defined” virtual system configuration, invoking the intrinsic References( ) CIM operation with parameter values set as follows:

   – The value of the ObjectName parameter refers to the instance of the CIM_VirtualSystemSettingData class that is the head element of the “State” virtual system configuration.

   – The value of the ResultClass parameter is set to “CIM_ElementSettingData”.

   The result of this step is a set of instances of the CIM_ElementSettingData association. From that set, the client selects the only instance where the IsDefault property has a value of 1 (Is Default). The value of the SettingData property of that instance refers to the instance of the

CIM_VirtualSystemSettingData class that is the head element of the “Defined” virtual system configuration.

3) The client resolves the CIM_VirtualSystemSettingDataComponent association to find instances of the CIM_ResourceAllocationSettingData class that represent resource allocation requests within the “Defined” virtual system configuration, invoking the intrinsic Associators( ) CIM operation with parameter values set as follows:

   – The value of the ObjectName parameter refers to the instance of the CIM_VirtualSystemSettingData class that is the head element of the “Defined” virtual system configuration.

   – The value of the AssocClass parameter is set to “CIM_VirtualSystemSettingDataComponent”.

   – The value of the ResultClass parameter is set to “CIM_ResourceAllocationSettingData”.

   The result of this step is a set of instances of the CIM_ResourceAllocationSettingData class that represent virtual resource allocation requests of the virtual system that are elements of the “Defined” virtual system configuration.

4) The client inspects the set of instances obtained in step 3), selecting only those instances where the ResourceType property has a value of 4 (Memory).

5) For each of the instances selected in step 4), the client then multiplies the of the VirtualQuantity property with 1024, yielding the amount of virtual memory requested through the inspected instance. The client builds the sum from the calculated values of all inspected instances with step 5).

NOTE: Subclause 7.1 requires the value of the AllocationUnits property to be “byte*2^10”. Alternatively the client might inspect the value of the AllocationUnits property in order to determine the unit.

Result: The client knows the amount of virtual memory requested for the virtual system.

In the example CIM representation shown in Figure 5, the client initially would know the instance of the CIM_ComputerSystem class tagged VS that represents the virtual system. From there, the client would follow the CIM_SettingsDefineState association to the instance of the CIM_VirtualSystemSettingData class tagged STA_VSSD. From there, the client would follow the CIM_ElementSettingData association where property IsDefault has a value of 1 (Is Default) to the instance of the CIM_VirtualSystemSettingData class tagged DEF_VSSD. Finally, from there the client would follow the CIM_VirtualSystemSettingDataComponent association to obtain the instance of the CIM_ResourceAllocationSettingData class tagged MEM_DEF that represents the only virtual memory resource allocation request for the virtual system.

The client would then multiply the value of the VirtualQuantity property with 1024, yielding a result of 1024*1048576 bytes or 1048576 KB requested for the virtual systems virtual memory.

9.2.3 Determine the Allocation Capabilities or Allocation Mutability

Assumption: Any of the following:

- The client knows a reference to an instance of the CIM_System class that represents a host system.
- The client knows a reference to an instance of the CIM_ResourcePool class that represents a memory resource pool.
- The client knows a reference to an instance of the CIM_ResourceAllocationSettingData class that represents a memory resource allocation request.
- The client knows a reference to an instance of the CIM_ResourceAllocationSettingData class that represents a memory resource allocation.

In the first two cases, this use case describes determining the related memory resource allocation capabilities; in the latter two cases, this use case describes determining the related mutability.

The sequence of activities is as follows:

1) The client resolves the CIM_ElementCapabilities association to find instances of the CIM_AllocationCapabilities class that represent allocation capabilities or mutability, invoking the intrinsic Associators( ) CIM operation with parameter values set as follows:
   - The value of the ObjectName parameter refers to the input instance.
   - The value of the AssocClass parameter is set to “CIM_ElementCapabilities”.
   - The value of the ResultClass parameter is set to “CIM_AllocationCapabilities”.

   The result of this step is a set of instances of the CIM_AllocationCapabilities class that represent allocation capabilities or mutability.
2) The client cycles through the set of instances of the CIM_AllocationCapabilities class obtained in step 1), fetching instances of the CIM_SettingsDefineCapabilities association that reference each of the instances from step 1) by invoking the intrinsic References( ) CIM operation with parameter values set as follows:
   - The value of the ObjectName parameter each cycle refers another instance of the CIM_AllocationCapabilities class from the result set of step 1).
   - The value of the ResultClass parameter is set to “CIM_SettingsDefineCapabilities”.

   The result of this step each time is a set of instances of the CIM_SettingsDefineCapabilities association.
3) For each of the instances obtained in step 2), the client inspects the values of the ValueRole and ValueRange properties to determine the type of limitation imposed by the instance of the CIM_ResourceAllocationSettingData class referenced by the value of the PartComponent property in that association instance, as follows:
   - A default setting is designated through a value of 0 (Default) for the ValueRole property and a value of 0 (Point) for the ValueRange property. A default setting does not apply for the description of mutability.
   - A minimum setting is designated through a value of 3 (Supported) for the ValueRole property and a value of 1 (Minimums) for the ValueRange property.
   - A maximum setting is designated through a value of 3 (Supported) for the ValueRole property and a value of 2 (Maximums) for the ValueRange property.
   - An increment setting is designated through a value of 3 (Supported) for the ValueRole property and a value of 3 (Increments) for the ValueRange property.
4) The client obtains for each of the instances obtained in step 2) the referenced instance of the CIM_ResourceAllocationSettingData class, invoking the intrinsic GetInstance( ) CIM operation with the value of the InstanceName parameter set to the value of the PartComponent property from the association instance. That value refers to the instance of the CIM_ResourceAllocationSettingData class that represents the capabilities setting.

   The result is the instance of the CIM_ResourceAllocationSettingData class with the values of all non-null numeric properties that describe the settings.

Result: The client knows the memory allocation capabilities of the system or the memory resource pool, or the mutability of a memory resource allocation request or a memory resource allocation.

In the example CIM representation shown in Figure 5, the capabilities of the system and the resource pool are represented by the instance of the CIM_AllocationCapabilities class tagged CAP that is referenced likewise through an instance of the CIM_ElementCapabilities association from the instance of the CIM_System class tagged HOST that represents the system and from the instance of the CIM_ResourcePool class tagged MEM_POOL that represents a memory resource pool. Four instances of the CIM_ResourceAllocationSettingData class tagged CAP_DEF, CAP_MIN, CAP_MAX, and CAP_INC describe the applicable default, minimum, maximum, and increment settings that describe the allocation capabilities. These instances are all associated through respectively parameterized instances of the CIM_SettingsDefineCapabilities association with the instance of the CIM_AllocationCapabilities class tagged CAP.

In the example CIM representation shown in Figure 5, the mutability of the memory resource allocation request is represented by the instance of the CIM_AllocationCapabilities class tagged DEF_MUT. The mutability of the memory resource allocation is represented by the instance of the CIM_AllocationCapabilities class tagged DEF_STA. Values of elements in the value sets of the SupportedAddStates[ ] and SupportedRemoveStates[ ] array properties indicate the virtual system state for which respective additions and removals or memory resource allocation requests or memory resource allocations are supported. For both instances of the CIM_AllocationCapabilities class, three instances of the CIM_ResourceAllocationSettingData class are associated through respectively parameterized instances of the CIM_SettingsDefineCapabilities association that describe minimum, maximum, and increment settings.

9.2.4 Determine the Default Memory Allocation Capabilities

Assumption: The client knows a reference to the instance of the CIM_System class that represents the host system.

The sequence of activities is as follows:

1) The client obtains instances of the CIM_ElementCapabilities association that reference the instance of the CIM_System class, invoking the intrinsic References( ) CIM operation with parameter values set as follows:
   - The value of the ObjectName parameter refers to the instance of the CIM_System class.
   - The value of the ResultClass parameter is set to “CIM_ElementCapabilities”.

   The result of this step is a set of instances of the CIM_ElementCapabilities association.
2) From the result set of step 1), the client drops those instances where the value set of the Characteristics[ ] array property does not contain an element with the value 2 (Default).

   The result of this step is a set of instances of the CIM_ElementCapabilities association that reference instances of the CIM_AllocationCapabilities class that represent the default allocation capabilities of the system for a number of resource types.
3) For each of the association instances obtained in step 2), the client obtains the instance of the CIM_AllocationCapabilities class that is referenced by the value of the Capabilities property in the respective association instance, invoking the intrinsic GetInstance( ) CIM operation with the value of the InstanceName parameter set to the value of the Capabilities property.

   The result of this step is a set of instances of the CIM_ResourceAllocationSettingData class that represent the system’s default allocation capabilities for a number of resource types.
4) From the result set of step 3), the client drops those instances where the value set of the ResourceType property is not 4 (Memory).

   The result of this step is one instance of the CIM_ResourceAllocationSettingData class that represents the system’s default allocation capabilities for the memory resource type. The client continues as in use case 9.2.3 step 2) in order to determine the set of instances of the

CIM_ResourceAllocationSettingData that represent the settings for the default memory resource allocation capabilities.

Result: The client knows the default memory allocation capabilities of the system.

In the example CIM representation shown in Figure 5, the default allocation capabilities for the memory resource type of the system are represented by the instance of the CIM_AllocationCapabilities class tagged CAP.

9.2.5 Determine the Default Memory Resource Pool

Assumption: The client knows a reference to the instance of the CIM_AllocationCapabilities class that represents the default memory resource allocation capabilities of the system; see 9.2.4.

The sequence of activities is as follows:

1) The client obtains instances of the CIM_ElementCapabilities association that reference the instance of the CIM_AllocationCapabilities class, invoking the intrinsic References( ) CIM operation with parameter values set as follows:

   – The value of the ObjectName parameter refers to the instance of the CIM_AllocationCapabilities class.

   – The value of the ResultClass parameter is set to “CIM_ElementCapabilities”.

   The result of this step is a set of instances of the CIM_ElementCapabilities association.

2) From the result set of step 1), the client drops those instances where the value set of the Characteristics[ ] array property does not contain an element with the value 2 (Default).

   The result of this step is a set of two instances of the CIM_ElementCapabilities association. One association instance references the instance of the CIM_ResourcePool class that represent the default memory resource pool, and one instance references the instance of the CIM_System class that represents the host system.

3) The client selects the instance of the CIM_ElementCapabilities association from the result of step 2) that references the instance of the CIM_ResourcePool class by comparing the value of the ManagedElement property against the known reference to the CIM_System class that represents the host system and dropping that association instance. The client uses the remaining association instance from the result set of step 2) to obtain the instance of the CIM_ResourcePool class that is referenced by the value of the ManagedElement property in that association instance, invoking the intrinsic GetInstance( ) CIM operation with the value of the InstanceName parameter set to the value of the ManagedElement property.

   The result of this step is the instance of the CIM_ResourcePool class that represents the system’s default memory resource pool.

Result: The client knows the default memory resource pool of the system.

In the example CIM representation shown in Figure 5, the default memory resource pool is represented by the instance of the CIM_ResourcePool class tagged MEM_POOL.

9.2.6 Obtain the Memory Pool with the Largest Unreserved Capacity

Assumption: The client knows a reference to the instance of the CIM_System class that represents the host system.

The sequence of activities is as follows:

1) The client resolves the CIM_HostedPool association to find instances of the CIM_ResourcePool class that represent resource pools hosted by the host system, invoking the intrinsic AssociatorNames( ) CIM operation with parameter values set as follows:
   – The value of the ObjectName parameter refers to the instance of the CIM_System class that represents the host.
   – The value of the AssocClass parameter is set to “CIM_HostedPool”.
   – The value of the ResultClass parameter is set to “CIM_ResourcePool”.

   The result of this step is a set of instances of the CIM_ResourcePool class that represent resource pools hosted by the host system.
2) The client selects from the result set of step 1) only those instances where the value of the ResourceType property is 4 (Memory).

   The result is a set of instances of the CIM_ResourcePool class that represent memory resource pools hosted by the host system.
3) The client inspects the value of the Capacity and the Reserved properties in all instances selected with step 2), and each time calculates the amount of unreserved memory capacity by subtracting the value of the Reserved property from the value of the Capacity property.
4) From all pools inspected in step 3), the client selects the one that has the largest free capacity.
5) The client checks the resource pool selected in step 4) for architectural limitations as expressed by the pool’s capabilities, applying use case 9.2.3.

Result: The client knows the memory resource pool with the largest unreserved capacity.

NOTE: The largest extent of memory actually available may be significantly smaller than indicated by the result because fragmentation may subdivide the amount of memory available into several smaller extents.

In the example CIM representation shown in Figure 5, the client initially would know the instance of the CIM_System class tagged HOST that represents the host system. From there, the client would follow the CIM_HostedPool association to instances of the CIM_ResourcePool class. Typically the association resolution would yield more than one instance, including instances that represent resource pools of other resource types, such that the client is required to select only those instances where the value of the ResourceType property is 4 (Memory). In Figure 5, there is only the instance of the CIM_ResourcePool class, tagged MEM_POOL, so the selection process is not required. From that instance, the client takes the value of the Capacity property and subtracts the value of the Reserved property (4194304 – 1048576) KB, yielding 3145728 KB as the maximum amount of memory potentially available from the pool.

10 CIM Elements

Table 2 lists CIM elements that are defined or specialized for this profile. Each CIM element shall be implemented as described in Table 2. The CIM Schema descriptions for any referenced element and its sub-elements apply.

Classes 7 (“Implementation”) and 8 (“Methods”) may impose additional requirements on these elements.

1279 **Table 2 – CIM Elements: Memory Resource Virtualization Profile**

Element	Requirement	Description
Classes		
CIM_AffectedJobElement	Conditional	See 10.1.
CIM_AllocationCapabilities (Capabilities)	Mandatory	See 10.2.
CIM_AllocationCapabilities (Mutability)	Optional	See 10.3.
CIM_Component (Memory Composition)	Optional	See 10.4.
CIM_Component (Resource Pool)	Optional	See 10.5.
CIM_ConcreteJob	Conditional	See 10.6.
CIM_ElementAllocatedFromPool	Mandatory	See 10.7.
CIM_ElementCapabilities (Capabilities)	Mandatory	See 10.8.
CIM_ElementCapabilities (Mutability)	Conditional	See 10.9
CIM_ElementCapabilities (Resource Pool)	Mandatory	See DSP1041.
CIM_ElementSettingData (Memory Pool)	Mandatory	See 10.10.
CIM_ElementSettingData (Memory Resource)	Mandatory	See 10.11.
CIM_HostedDependency	Optional	See 10.12.
CIM_HostedResourcePool	Mandatory	See DSP1041.
CIM_HostedService	Mandatory	See DSP1041.
CIM_Memory (Host System)	Conditional	See 10.13.
CIM_Memory (Virtual System)	Mandatory	See 10.14.
CIM_RegisteredProfile	Mandatory	See 10.15.
CIM_ResourceAllocationFromPool	Optional	See 10.16.
CIM_ResourceAllocationSettingData	Mandatory	See 10.17.
CIM_ResourcePool	Mandatory	See 10.18.
CIM_ResourcePoolConfigurationCapabilities	Mandatory	See 10.19.
CIM_ResourcePoolConfigurationService	Mandatory	See DSP1041.
CIM_SettingsDefineState	Mandatory	See 10.20.
CIM_ServiceAffectsElement	Mandatory	See 10.21.
CIM_SystemDevice (Virtual Memory)	Mandatory	See 10.22.
CIM_SystemDevice (Host Memory)	Optional	See 10.23.
Indications		
None defined		

1280 ## 10.1 CIM_AffectedJobElement

1281 The support of the CIM_AffectedJobElement class is conditional.

1282 Conditional Requirement: The CIM_AffectedJobElement association shall be supported if asynchronous
1283 operations for resource pool management are supported; see 7.2.2.8.4.

If the CIM_AffectedJobElement association is supported, instances of the CIM_AffectedJobElement association shall associate an instance of the CIM_ConcreteJob class that represents an asynchronous memory resource pool management task and all of the following:

- the instance of the CIM_ResourcePool class that represents a memory resource pool that is affected by the asynchronous memory resource pool management task
- the instance of the CIM_Memory class that represents host memory that is affected by the asynchronous memory resource pool management task

Table 3 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 3 – Association: CIM_AffectedJobElement

Elements	Requirement	Notes
AffectedElement	Mandatory	**Key:** Value shall reference an instance of the CIM_ResourcePool class or the CIM_Memory class. **Cardinality: ***
AffectingElement	Mandatory	**Key:** Value shall reference the instance of the CIM_ConcreteJob class. **Cardinality:** *

10.2 CIM_AllocationCapabilities (Capabilities)

See 7.3.1.1 for detailed implementation requirements for this class if it is used for the representation of memory resource allocation capabilities of systems or of memory resource pools.

Table 4 lists the requirements for elements of this class in this case. These requirements are in addition to those specified in the CIM Schema and in DSP1043.

Table 4 – Class: CIM_AllocationCapabilities (Memory Allocation Capabilities)

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ResourceType	Mandatory	Value shall be 4 (Memory).
OtherResourceType	Mandatory	Value shall be NULL.
RequestTypesSupported	Mandatory	See 7.3.1.1.2.
SharingMode	Mandatory	See 7.3.1.1.3.
SupportedAddStates[ ]	Mandatory	Value shall be NULL.
SupportedRemoveStates[ ]	Mandatory	Value shall be NULL.

10.3 CIM_AllocationCapabilities (Mutability)

The support of the CIM_AllocationCapabilities class for the representation of the mutability of memory resource allocation requests and memory resource allocations is optional.

If the CIM_AllocationCapabilities class is supported for the representation of the mutability of memory resource allocation requests and memory resource allocations, see 7.3.2.2 for detailed implementation requirements.

Table 5 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1043.

Table 5 – Class: CIM_AllocationCapabilities (Memory Allocation Mutability)

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ResourceType	Mandatory	Value shall be 4 (Memory).
OtherResourceType	Mandatory	Value shall be NULL.
RequestTypesSupported	Mandatory	See 7.3.2.2.1.
SharingMode	Mandatory	See 7.3.2.2.2.
SupportedAddStates[ ]	Optional	See 7.3.2.2.3.
SupportedRemoveStates[ ]	Optional	See 7.3.2.2.4.

10.4 CIM_Component (Memory Composition)

The support of the CIM_Component association for the representation of memory compositions is optional.

If the CIM_Component association is supported for the representation of memory compositions that are composed of other memory extents, instances of an association that is based on the CIM_Component association shall associate an instance of the CIM_Memory class that represents the memory composition and any instance of the CIM_Memory class that represent composing memory extents. If an implementation does not implement a more specific association based on the CIM_Component association, the CIM_ConcreteComponent association should be implemented.

NOTE: The CIM_Component association is abstract; therefore it cannot be directly implemented. On the other hand, clients may directly follow abstract associations.

Table 6 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema.

Table 6 – Association: CIM_Component (Memory Resource)

Elements	Requirement	Notes
GroupComponent	Mandatory	**Key:** Value shall reference the instance of the CIM_Memory class that represents the memory composition. **Cardinality:** 0..1
PartComponent	Mandatory	**Key:** Value shall reference an instance of the CIM_Memory class that represents a composing memory extent. **Cardinality:** *

10.5 CIM_Component (Resource Pool)

The support of the CIM_Component association for the representation of memory extents that are aggregated by resource pools is optional.

If the CIM_Component association is supported for the representation of memory extents that are aggregated by resource pools, instances of an association that is based on the CIM_Component association shall associate an instance of the CIM_ResourcePool class that represents a primordial memory resource pool and any instance of the CIM_Memory class that represents host memory that is aggregated into the pool. If an implementation does not implement a more specific association based on the CIM_Component association, the CIM_ConcreteComponent association should be implemented.

NOTE: The CIM_Component association is abstract; therefore it cannot be directly implemented. On the other hand, clients may directly follow abstract associations.

Table 7 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 7 – Association: CIM_Component (Resource Pool)

Elements	Requirement	Notes
GroupComponent	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourcePool class that represents a memory resource pool. **Cardinality:** 0..1
PartComponent	Mandatory	**Key:** Value shall reference an instance of the CIM_Memory class that represents host memory aggregated into the pool. **Cardinality:** *

10.6 CIM_ConcreteJob

The support of the CIM_ConcreteJob class is conditional.

Conditional Requirement: The CIM_ConcreteJob class shall be supported if asynchronous operations for resource pool management are supported; see 7.2.2.8.4.

If the CIM_ConcreteJob is supported, instances of the CIM_ConcreteJob class shall represent asynchronous memory resource pool management tasks.

Table 8 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 8 – Class: CIM_ConcreteJob

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
DeleteOnCompletion	Mandatory	Value shall be TRUE.
ElementName	Mandatory	Value shall conform to pattern ".*".
ErrorCode	Mandatory	None
ErrorDescription	Mandatory	None
JobStatus	Mandatory	None
JobState	Mandatory	None

10.7 CIM_ElementAllocatedFromPool

An instance of the CIM_ElementAllocatedFromPool association shall associate an instance of the CIM_ResourcePool class that represents a memory resource pool with each instance of the CIM_Memory class that represents virtual memory resulting from a memory resource allocation out of the pool.

Table 9 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 9 – Association: CIM_ElementAllocatedFromPool

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourcePool class that represents a memory resource pool. **Cardinality:** 1
Dependent	Mandatory	**Key:** Value shall reference the instance of the CIM_Memory class that represents virtual memory resulting from a memory allocation from the pool. **Cardinality:** *

10.8 CIM_ElementCapabilities (Capabilities)

Instances of the CIM_ElementCapabilities association shall associate all of the following:

- the instance of the CIM_System class that represents the host system with each instance of the CIM_AllocationCapabilities class that represents memory allocation capabilities of the host system
- an instance of the CIM_ResourcePool that represents a memory resource pool with each instance of the CIM_AllocationCapabilities class that represents memory allocation capabilities of the memory resource pool

Table 10 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1043.

Table 10 – Association: CIM_ElementCapabilities (Capabilities)

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** Host: See 7.3.1.2 and 7.3.1.3. Pool: See 7.3.1.4 and 7.3.1.5. **Cardinality:** *
Capabilities	Mandatory	**Key:** Host: See 7.3.1.2 and 7.3.1.3. Pool: See 7.3.1.4 and 7.3.1.5. **Cardinality:** *
Characteristics	Mandatory	Host: See 7.3.1.2 and 7.3.1.3. Pool: See 7.3.1.4 and 7.3.1.5.

10.9 CIM_ElementCapabilities (Mutability)

The support of the CIM_ElementCapabilities association for the representation of the mutability of a memory resource allocation request or a memory resource allocation is conditional.

Conditional Requirement: The support is required if the CIM_AllocationCapabilities class is supported for the representation of the mutability of memory resource allocation requests or memory resource allocations; see 10.3.

If the CIM_ElementCapabilities association is supported for the representation of the mutability of a memory resource allocation request or a memory resource allocation, an instance of the CIM_ElementCapabilities association shall associate an instance of the CIM_ResourceAllocationSettingData class that represents a memory resource allocation or memory resource allocation request with each instance of the CIM_AllocationCapabilities class that represents the mutability of the memory resource allocation or memory resource allocation request.

Table 11 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1043.

Table 11 – Association: CIM_ElementCapabilities (Mutability)

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourceAllocationSettingData class. **Cardinality:** *
Capabilities	Mandatory	**Key:** Value shall reference the instance of the CIM_AllocationCapabilities class. **Cardinality:** *
Characteristics[ ]	Mandatory	Value shall be { 3 (Current) }.

10.10 CIM_ElementSettingData (Memory Resource Pool)

An instance of the CIM_ElementSettingData association shall associate an instance of the CIM_ResourcePool class that represents a concrete memory resource pool and the instance of the CIM_ResourceAllocationSettingData class that represents the memory resource allocation that describes the allocation of the concrete resource pool from another resource pool.

Table 12 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 12 – Association: CIM_ElementSettingData (Memory Resource Pool)

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourcePool class that represents the concrete memory resource pool. **Cardinality:** 0..1
SettingData	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourceAllocationSettingData class that represents the memory resource allocation. **Cardinality:** 0..1

10.11 CIM_ElementSettingData (Memory Resource)

An instance of the CIM_ElementSettingData association shall associate an instance of the CIM_ResourceAllocationSettingData class that represents a memory resource allocation and the instance of the CIM_ResourceAllocationSettingData class that represents the corresponding memory resource allocation request.

Table 13 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 13 – Association: CIM_ElementSettingData (Memory Resource)

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourceAllocationSettingData class that represents the memory resource allocation. **Cardinality:** 1
SettingData	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourceAllocationSettingData class that represents the memory resource allocation request. **Cardinality:** 0..1
IsDefault	Mandatory	Value shall be 1 (Is Default).
IsCurrent	Mandatory	Unspecified.
IsNext	Mandatory	Unspecified.

10.12 CIM_HostedDependency

The support of the CIM_HostedDependency association is optional.

If the CIM_HostedDependency association is supported, an instance of the CIM_HostedDependency association shall associate an instance of the CIM_Memory class that represents virtual memory with the instance of the CIM_Memory class that represents host memory that is dedicated for the support of the virtual memory.

Table 14 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 14 – Association: CIM_HostedDependency

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** Value shall reference the instance of the CIM_Memory class that represents host memory. **Cardinality:** 0..1
Dependent	Mandatory	**Key:** Value shall reference the instance of the CIM_Memory class that represents virtual memory. **Cardinality:** 0..1

10.13 CIM_Memory (Host System)

The support of the CIM_Memory class for the representation of host memory is conditional.

Conditional Requirement: The support is required if the CIM_SystemDevice association is supported for the representation of host memory; see 10.23.

Table 15 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1026 if that is implemented.

Table 15 – Class: CIM_Memory (Host System)

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	**Key**
CreationClassName	Mandatory	**Key**
SystemName	Mandatory	**Key**
Name	Mandatory	**Key**
EnabledState	Mandatory	Unspecified
RequestedState	Mandatory	Unspecified
StartingAddress	Mandatory	Unspecified
EndingAddress	Mandatory	Unspecified

10.14 CIM_Memory (Virtual System)

See 7.2.5 for detailed implementation requirements for this class if it is used for the representation of virtual memory or virtual memory composition.

Table 16 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1026 if that is implemented.

Table 16 – Class: CIM_Memory (Virtual System)

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	**Key**
CreationClassName	Mandatory	**Key**
SystemName	Mandatory	**Key**
Name	Mandatory	**Key**
EnabledState	Mandatory	Unspecified
RequestedState	Mandatory	Unspecified
StartingAddress	Mandatory	Unspecified
EndingAddress	Mandatory	Unspecified

10.15 CIM_RegisteredProfile

The use of the CIM_RegisteredProfile class is specified by DSP1033.

Table 17 lists the requirements for elements of this class. These requirements are in addition to those specified in DSP1033.

Table 17 – Class: CIM_RegisteredProfile

Elements	Requirement	Notes
RegisteredOrganization	Mandatory	Value shall be set to 2 (DMTF).
RegisteredName	Mandatory	Value shall be set to “Memory Resource Virtualization”.
RegisteredVersion	Mandatory	Value shall be set to the version of this profile: “1.0.0”.

10.16 CIM_ResourceAllocationFromPool

The support of the CIM_ResourceAllocationFromPool association is optional.

If the CIM_ResourceAllocationFromPool association is supported, an instance of the CIM_ResourceAllocationFromPool association shall associate an instance of the CIM_ResourcePool class that represents a memory resource pool with each instance of the CIM_ResourceAllocationSettingData class that represents a memory resource allocation from the pool.

Table 18 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 18 – Association: CIM_ResourceAllocationFromPool

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourcePool class that represents a memory resource pool. **Cardinality:** 0..1
Dependent	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourceAllocationSettingData class that represents a memory resource allocation from the pool. **Cardinality:** *

10.17 CIM_ResourceAllocationSettingData

See 7.3.2.2 for detailed implementation requirements for this class.

Table 19 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

1435 **Table 19 – Class: CIM_ResourceAllocationSettingData**

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**; see DSP1041.
ResourceType	Mandatory	Value shall be 4 (Memory).
OtherResourceType	Mandatory	Value shall be NULL.
ResourceSubType	Optional	See DSP1041.
PoolID	Mandatory	See 7.2.4.2.
ConsumerVisibility	Optional	See 7.2.4.3.
HostResource[ ]	Optional	See 7.2.4.4.
AllocationUnits	Mandatory	See 7.1.
VirtualQuantity	Mandatory	See 7.2.4.5.
Reservation	Optional	See 7.2.4.6.
Limit	Optional	See 7.2.4.7.
Weight	Optional	See 7.2.4.8.
AutomaticAllocation	Optional	See DSP1041.
AutomaticDeallocation	Optional	See DSP1041.
Parent	Optional	See 7.2.4.9.
Connection[ ]	Optional	See 7.2.4.10.
MappingBehavior	Optional	See 7.2.4.11.

1436 ## 10.18 CIM_ResourcePool

1437 Instances of the CIM_ResourcePool class shall represent memory resource pools.

1438 Table 20 lists the requirements for elements of this class. These requirements are in addition to those
1439 specified in the CIM Schema and in DSP1041.

1440 **Table 20 – Class: CIM_ResourcePool**

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ElementName	Optional	See DSP1041.
PoolID	Mandatory	See 7.2.2.1.
Primordial	Mandatory	See 7.2.2.2.
Capacity	Conditional	See 7.2.2.5.
Reserved	Optional	See 7.2.2.4.
ResourceType	Mandatory	Value shall be 4 (Memory).
OtherResourceType	Mandatory	Value shall be NULL.
ResourceSubType	Optional	See DSP1041.
AllocationUnits	Mandatory	See 7.1.

10.19 CIM_ResourcePoolConfigurationCapabilities

An instance of the CIM_ResourcePoolConfigurationCapabilities class shall represent the capabilities of a memory resource pool configuration service.

Table 21 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 21 – Class: CIM_ResourcePoolConfigurationCapabilities

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
AsynchronousMethodsSupported[ ]	Mandatory	See 7.2.2.8.
SynchronousMethodsSupported[ ]	Mandatory	See 7.2.2.8.

10.20 CIM_SettingsDefineState

An instance of the CIM_SettingsDefineState association shall associate an instance of the CIM_Memory class that represents virtual memory and the instance of the CIM_ResourceAllocationSettingData class that represents the memory resource allocation that yields the virtual memory.

Table 22 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 22 – Association: CIM_SettingsDefineState

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** Value shall reference an instance of the CIM_Memory class. **Cardinality:** 0..1
SettingData	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourceAllocationSettingData class. **Cardinality:** 0..1

10.21 CIM_ServiceAffectsElement

An instance of the CIM_ServiceAffectsElement association shall associate an instance of the CIM_ResourcePoolConfigurationService class that represents a memory resource pool configuration service and each instance of the CIM_ResourcePool class that represents a memory resource pool that is configurable through the service.

Table 23 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 23 – Association: CIM_ServiceAffectsElement

Elements	Requirement	Notes
AffectedElement	Mandatory	**Key:** Value shall reference an instance of the CIM_ResourcePool class. **Cardinality:** *

Elements	Requirement	Notes
AffectingElement	Mandatory	**Key:** Value shall reference the instance of the CIM_ResourcePoolConfigurationService class. **Cardinality:** 1

10.22 CIM_SystemDevice (Virtual Memory)

An instance of the CIM_SystemDevice association shall associate the instance of the CIM_ComputerSystem class that represents a virtual system and each instance of the CIM_Memory class that represents virtual memory in scope of the virtual system.

Table 24 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 24 – Association: CIM_SystemDevice (Virtual Memory)

Elements	Requirement	Notes
GroupComponent	Mandatory	**Key:** Value shall reference an instance of the CIM_System class. **Cardinality:** 1
PartComponent	Mandatory	**Key:** Value shall reference the instance of the CIM_Memory class. **Cardinality:** *

10.23 CIM_SystemDevice (Host Memory)

Support of the CIM_SystemDevice association for the representation of host memory is optional; see 7.2.1.

NOTE: Support is mandatory if DSP1026 is implemented for the host system.

If the CIM_SystemDevice association is supported for the representation of host memory, an instance of the CIM_SystemDevice association shall associate the instance of the CIM_System class that represents the scoping host system and each instance of the CIM_Memory class that represents host memory in scope of the scoping host system.

Table 25 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema, in DSP1041, and in DSP1026 if that is implemented.

Table 25 – Association: CIM_SystemDevice (Host Memory)

Elements	Requirement	Notes
GroupComponent	Mandatory	**Key:** Value shall reference an instance of the CIM_System class. **Cardinality:** 1
PartComponent	Mandatory	**Key:** Value shall reference the instance of the CIM_Memory class. **Cardinality:** *

ANNEX A
(Informative)

Change Log

Version	Date	Description
1.0.0	2009-07-14	DMTF Standard Release