Document Number: DSP1044

Date: 2010-04-22

Profile Version: 1.0.0

Processor Resource Virtualization Profile

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

84 ## Figures

95

96 ## Tables

Foreword

This profile was prepared by the Server Virtualization Partitioning and Clustering workgroup of the DMTF.

The SVPC work group acknowledges the following people for their contributions to the development this profile.

Editors:

- Michael Johanssen – IBM
- Lawrence Lamers – VMware Inc.
- Carl Waldspurger - VMware Inc.

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

Introduction

The information in this specification should be sufficient for a provider or consumer of this data to unambiguously identify the classes, properties, methods, and values that shall be instantiated and manipulated to represent and manage a basic server and subsystems that are modeled using the DMTF Common Information Model (CIM) core and extended model definitions.

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.

Processor Resource Virtualization Profile

1 Scope

This profile is a component profile that extends the management capabilities of the specialized profiles by adding the support to represent and manage the allocation of processor resources to virtual systems.

2 Normative references

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

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

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

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

DMTF DSP1022, *CPU profile* 1.0
http://www.dmtf.org/standards/published_documents/DSP1022_1.0.pdf

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

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

DMTF DSP1042, *System Virtualization profile* 1.0
http://www.dmtf.org/standards/published_documents/DSP1042_1.0.pdf

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

DMTF DSP1057, *Virtual System profile* 1.0
http://www.dmtf.org/standards/published_documents/DSP1057_1.0.pdf

DMTF DSP1059, *Generic Device Resource Virtualization profile* 1.0
http://www.dmtf.org/standards/published_documents/DSP1059_1.0.pdf

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

3 Terms and definitions

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

3.1
can
used for statements of possibility and capability, whether material, physical, or causal

3.2
cannot
used for statements of possibility and capability, whether material, physical, or causal

3.3
conditional
indicates requirements to be followed strictly to conform to the document when the specified conditions are met

3.4
mandatory
indicates requirements to be followed strictly 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 "Referenced Profiles" table

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

3.10
shall not
indicates requirements to be followed strictly 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 or operation

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 Processor Resource Virtualization profile

3.17
host processor resource
host processor resources are processor devices or computing resource contained by the host system that may be allocated with either exclusive or shared access to provide processing resources to a processor resource pool or a virtual system.

3.18
host system
The scoping system containing resources that may be allocated and/or virtualized.

3.19
processor resource
a processor device or computing resource as seen by a consumer

3.20
processor resource allocation
the allocation of a processor resource from a processor resource pool to a virtual system

3.21
processor resource allocation request
a request for a processor resource allocation

3.22
processor resource pool
a resource pool that represents processor resources available for processor resource allocation

3.23
processor resource pool configuration service
a configuration service that supports the addition or removal of host storage processor resources to or from a processor resource pool, and the creation or deletion of concrete subpools of a processor resource pool

3.24
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, and container.#

3.25
virtual processor
the instantiation of the allocated host processor resources that is exposed to a virtual system via a logical processor device.

4 Symbols and abbreviated terms

The following abbreviations are used in this document.

4.1
CIM
Common Information Model

4.2
CIMOM
CIM object manager

4.3
RASD
CIM_ResourceAllocationSettingData

4.4
VS
virtual system

5 Synopsis

Profile Name: Processor Resource Virtualization

Version: 1.0.0

Organization: DMTF

CIM schema version: 2.21

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 processors.

Table 1 lists other profiles that this profile depends on, or that may be used in context of this profile.

281 **Table 1 – Related profiles**

Profile Name	Organization	Version	Relationship	Description
Resource Allocation	DMTF	*1.1*	Specializes	The profile that adds the capability to represent the allocation of resources to consumers. See DSP1041.
Allocation Capabilities	DMTF	*1.0*	Specializes	The profile that describes the default property values, supported property values, and range of property values for a resource allocation request. See DSP1043.
Profile Registration	DMTF	*1.0*	Mandatory	The profile that specifies registered profiles. See DSP1033.
CPU	DMTF	*1.0*	Optional	The profile that adds the capability to represent processors in a managed system. See 7.2. See DSP1022.

282 The CPU Profile lists additional related DMTF management profiles; these relationships are not further
283 specified in this profile.

284 6 Description

285 This clause introduces the management domain addressed by this profile, and outlines the central
286 modeling elements established for representation and control of the management domain.

287 6.1 General

288 In computer virtualization systems, virtual computer systems are composed of component virtual
289 resources. This profile specifies the allocation and management of host processor resources in support of
290 virtual processors. From an operating system or application viewpoint virtual processors are functionally
291 equivalent to "physical" processors.

292 This profile applies the resource virtualization pattern defined in DSP1041 and the allocation capabilities
293 pattern defined in DSP1043 to enable the management of processor resources that are allocated to
294 virtual systems. This profile defines additional CIM elements and constraints beyond those defined in the
295 specialized profiles. Optionally implementations may implement DSP1022 to represent host processors.

296 6.2 Processor resource virtualization class schema

297 Figure 1 represents the class schema of this profile. It outlines the elements that are adapted by this
298 profile. For simplicity, the prefix CIM_ has been removed from the name of the classes.

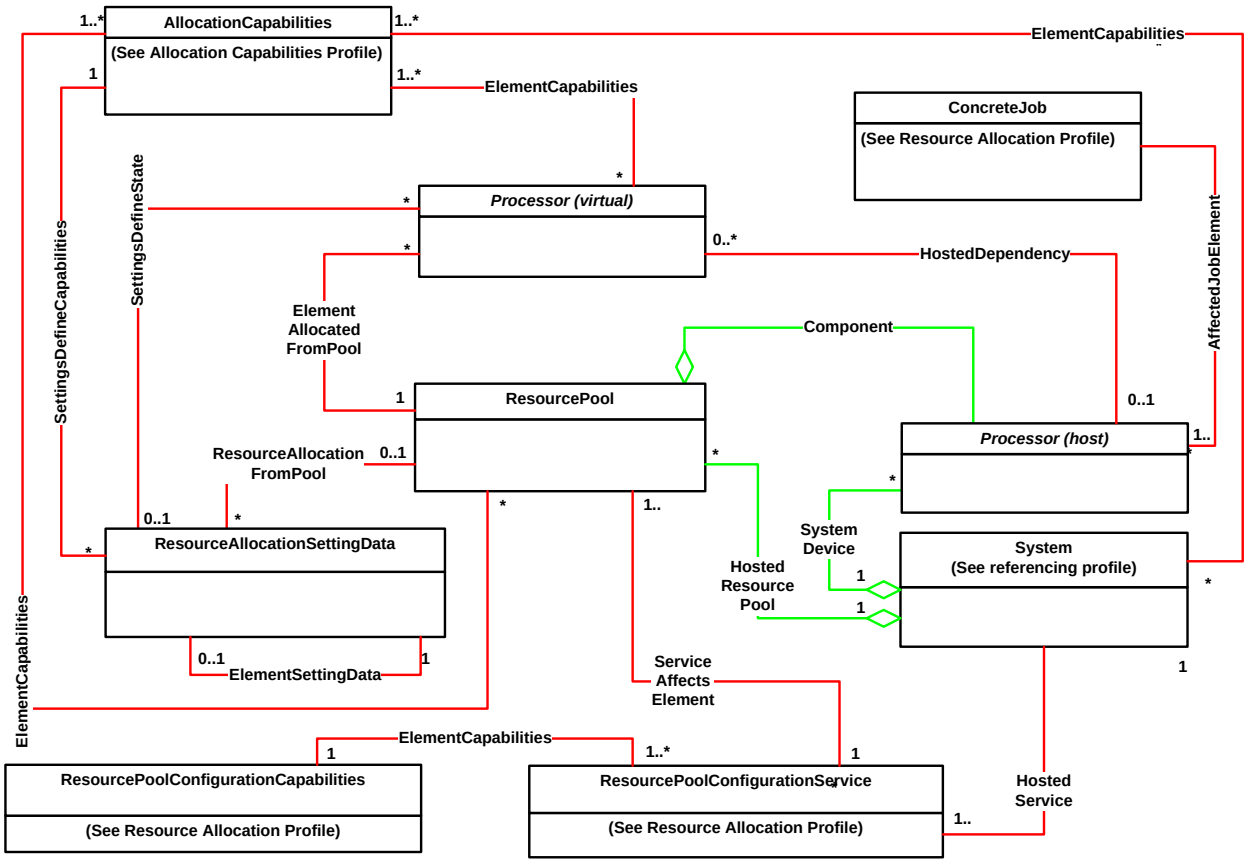


Figure 1 – Processor Resource Virtualization Profile: Class Diagram

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

- The CIM_ResourcePool class models resource pools for processor resources. Host processor resources are allocated from their resource pool and used to create the virtual processors for virtual systems
- The CIM_Component association models the relationship between processor resource pools and host processors as components of the resource pools
- The CIM_ElementAllocatedFromPool association models hierarchies of processor resource pools and modeling the relationship of processor resource pools and the virtual processors allocated from those
- The CIM_HostedResourcePool association models the hosting dependency between a processor resource pool and its host system. A host system supports at least one processor resource pool
- The CIM_Processor class models the following kinds of processors:
  - processors as a device in the scope of a system, as modeled by the CIM_SystemDevice association
  - processors as a result of a processor resource allocation from a resource pool, as modeled by the CIM_ElementAllocatedFromPool association
  - processors as a component within processor resource pools, as modeled through the CIM_Component association

- The CIM_ResourceAllocationSettingData class models processor resource allocations or processor resource allocation requests
- The CIM_AllocationCapabilities class and the CIM_ElementCapabilities association model
  - the processor resource allocation capabilities of host systems
  - the processor resource allocation capabilities of processor resource pools
  - the mutability of existing processor resource allocations
- The CIM_SettingsDefineCapabilities association models the relation between processor resource allocation capabilities and the settings that define these capabilities
- The CIM_ResourcePoolConfigurationService models configuration services for processor resource pools and the CIM_ResourcePoolConfigurationCapabilities class modeling their capabilities
- The CIM_ConcreteJob class and the CIM_AffectedJobElement association models asynchronous management tasks initiated through memory resource pool configuration services

In general, any mention of a class in this document means the class itself or its subclasses. For example, a statement such as "an instance of the CIM_Processor class" implies an instance of the CIM_Processor class or a subclass of the CIM_Processor class.

6.3 Resource pools

This profile applies the concept of resource pools defined in the DSP1041 to the resource type 3 (Processor).

6.3.1 General

This profile uses processor resource pools as the focal point for processor allocations. A processor resource pool represents an aggregate amount of processing capacity, and keeps track of the amount of processor capacity that has been allocated to consumers such as virtual systems. A resource pool also defines the scope in which relative weights are interpreted. A resource pool represents a part or all of the aggregated processing power of a virtualization platform in an abstract sense, that is, it represents the sum of the processing power of the host resources aggregated into the resource pool and is expressed in allocation units such as MHz, percentage, or count of processors.

Note that the resource type of a resource pool governs the type of the resources that are allocated from the resource pool. Opposed to that the resource type of the resources that are aggregated by the resource pool may differ from the resource type of the pool. For example, a resource pool with a resource type of 3 (Processor) supports the allocation of virtual processors. However, the resources that are aggregated by that resource pool may be of a different type; for example, that resource pool might aggregate processor fragments, or it might simply represent a certain amount of processing power without representing individual processors.

6.3.2 Representation of host resources

A processor resource pool represents an aggregated amount of processing power provided by host resources that enables the allocation of virtual processors. However the explicit representation of the host resources aggregated by a resource pool is optional: In some cases implementations may explicitly represent the host resources such as for example host processors. In other cases implementations may choose not to explicitly represent the host resources aggregated by a resource pool, that is, an implementation may choose to model a resource pool as the sole model element that represents host processing capacity for the support of (allocated) virtual processors, but not detail the host resources that provide that processing capacity.

The processor resources of a host system can be represented by a single processor resource pool, with capacity equal to the processing capacity of the host computer system. The processor resources of a host computer system may be represented by multiple resource pools, providing more flexible control over the allocation of processor resources to virtual systems.

6.3.3 Hierarchies of processor resource pools

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

6.4 Resource allocation

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

6.4.1 Processor resource allocation request

The processor requirements of a virtual system are defined as part of the "defined" virtual system configuration; see DSP1057 for a definition of the "defined" virtual system configuration. The "defined" virtual system configuration contains processor resource allocation requests represented as RASD instances.

6.4.2 Processor Resource allocation

As a virtual system is activated (or instantiated), one or more virtual processors need to be allocated as requested by processor resource allocation requests in the virtual system definition. Processor resource allocations are represented as RASD instances in the "state" virtual system configuration. The number of discrete processors exposed to the virtual system is specified by the value of the VirtualQuantity property.

The central properties describing a processor resource allocation request or a processor resource allocation are Reservation, Limit, and Weight.

The values of both the Reservation and Limit properties are specified in allocation units as expressed by the value of the AllocationUnits property. Possible allocation units are a frequency (i.e., "Hertz" or "MHz"), a percentage (with respect to some base value such as the sum of all available processing power), or a count (expressing a number of processors or processor fractions to be allocated).

The value of the Reservation property specifies a lower bound on the quantity of host processor resources available for the virtual computer system that are guaranteed to be available for use. If a virtual computer system does not consume its full allocation of reserved resources, the host system may allow its unused portion to be utilized by other virtual computer systems. The value of the Limit property specifies a limit or upper bound on the quantity of host processor resources that may be consumed by the virtual computer system. A limit is an artificial cap that may not be exceeded, even if otherwise-idle host processor resources are available.

The value of the Weight property specifies the relative importance of the set of allocated virtual processors, and is expressed in abstract numeric units. A virtual system is entitled to consume host processor resources at a rate that is directly proportional to the value of the Weight property.

6.4.3 Virtual processors

A virtual processor is the instantiation of allocated processor resources that is exposed to a virtual system through a logical processor device; it is the result of the processor resource allocation based on a processor resource allocation request. A virtual processor may be realized using techniques such as time sharing, but may also be a host processor that is directly passed through to the virtual system.

A virtual processor is represented by a CIM_Processor instance that is part of the virtual system representation.

6.4.4 Dedicated processors

A dedicated host processor is a processor owned by the host system that is exclusively reserved for support of a virtual processor of a particular virtual system.

6.4.5 Consumption of host processing power

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

6.4.5.1 Statically controlled processor resource consumption

With statically controlled processor resource consumption, a particular processor resource allocation request is granted only if the amount of host processing resource available in the addressed processor resource pool is at least as large as the amount requested. This approach is called admission control. Each successfully allocated processor resource reduces the amount of processing resources available from the pool, respectively. In the CIM representation of the processor resource pool, the amount of processing power assigned to consumers from the pool is visible through the value of the Reserved property. With admission control the processing capacity represented by a particular resource pool remains larger than the sum of all reservations out of that pool. Admission control checks are typically performed for the following operations: creating a resource pool, powering on a virtual system, and modifying the resource allocation settings for either a resource pool or a running virtual system.

6.4.5.2 Dynamically controlled processor resource consumption

With dynamically controlled processor resource consumption, the amount of host processing resources allocated in support of a virtual processor is not a constant value but varies significantly over time, depending on factors like the processing requirements of the software executed within the virtual system or the processing power consumption of other virtual machines.

Further, with dynamically controlled processor resource consumption, the amount of processing power managed through a processor resource pool conceptually may be considered as unlimited, such that no admission control is performed at the time virtual processors are initially allocated (usually at virtual system activation time). Of course, this approach may result in an over commitment situation where the host system experiences processing power shortages at a later point in time, such that ultimately the host system is no longer able to support the sum of processing power requests of all hosted virtual systems.

7 Implementation

This clause provides normative requirements related to the arrangement of instances and properties of instances for implementations of this profile.

7.1 Allocation units

This subclause details requirements for the unit of measurement that applies to the specification of processor resource allocations.

7.1.1 General

Processor resource allocations and processor resource allocation requests shall be expressed through DSP0004 programmatic units using one these base units: "hertz", "percent" or "count" .

7.1.2 Use of the base unit "hertz"

If the base unit of "hertz" is used, the following provisions apply:

- CIM_Processor class
  - MaxClockSpeed property: is expressed in MHz as defined in the CIM Schema
  - CurrentClockSpeed property: is expressed in MHz as defined in the CIM Schema
- CIM_ResourceAllocationSettingData class
  - AllocationUnits property: Value shall use a base unit of "hertz”.
  - Reservation property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.
  - Limit property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.
- CIM_ResourcePool class
  - AllocationUnits property: Value shall use a base unit of "hertz”.
  - Capacity property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.
  - Reserved property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.

7.1.3 Use of the base unit "percent"

If the base unit of "percent" is used, the following provisions apply:

- CIM_Processor class
  - MaxClockSpeed: Value is expressed in MHz as defined in the CIM Schema
  - CurrentClockSpeed: Value is expressed in MHz as defined in the CIM Schema
- CIM_ResourceAllocationSettingData class
  - AllocationUnits property: Value shall be "percent"
  - Reservation property: Value shall be expressed in percent, stating a minimum percentage of the processing power as requested from the resource pool identified by the RASD instance.
  - Limit property: Value shall be expressed in percent, stating a maximum percentage of the processing power to be provided by the resource pool identified by the RASD instance..
- CIM_ResourcePool class
  - AllocationUnits property: Value shall be "percent"
  - Capacity property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.
  - Reserved property: Value shall be expressed in percent, stating the reserved percentage of processing power presently allocated from the pool.

7.1.4 Use of the base unit "count"

If the base unit of "count" is used the following provisions apply:

- CIM_Processor class
  - MaxClockSpeed: Value is expressed in MHz as defined in the CIM Schema

- CurrentClockSpeed: Value is expressed in MHz as defined in the CIM Schema

- CIM_ResourceAllocationSettingData class
  - AllocationUnits property: Value shall use a base unit of "count", the counted items shall be processors. For example, a unit of a tenth of a processor can be expressed using the value “count*0.1”.
  - Reservation property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.
  - Limit property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.
- CIM_ResourcePool class
  - AllocationUnits property: Value shall be use a base unit of "count" the counted items shall be processors. For example, a unit of a tenth of a processor can be expressed using the value “count*0.1”.
  - Capacity property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.
  - Reserved property: Value shall be expressed in the unit expressed by the value of the AllocationUnits property.

7.2 Host resources

The implementation of the representation of host processor resources is optional.

If the representation of host processors is implemented, the provisions in this subclause apply.

Each host processor shall be represented by exactly one CIM_Processor instance that is associated with the CIM_System instance that represents the host system through an instance of the CIM_SystemDevice association.

Implementations may implement DSP1022 for host processors.

7.3 Resource pools

This subclause adapts the CIM_ResourcePool class for the representation of processor resource pools.

7.3.1 ResourceType property

The value of the ResourceType property shall be 3 (Processor).

7.3.2 ResourceSubType property

The implementation of the ResourceSubType property is optional.

If the ResourceSubType property is implemented, the provisions in this subclause apply.

The value of the ResourceSubType property shall designate a resource subtype. The format of the value shall be as follows: "<org-id>:<org-specific>". The <org-id> part shall identify the organization that defined the resource subtype value; the <org-specific> part shall uniquely identify a resource subtype within the set of subtype defined by the respective organization.

7.3.3 Primordial property

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

7.3.4 PoolID property

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

7.3.5 Reserved property

The implementation of the Reserved property is optional.

If the Reserved property is implemented, its value shall reflect the amount of host processing resource that is actually reserved from the resource pool, in units as expressed by the value of the AllocationUnits property (see 7.3.7) .

7.3.6 Capacity property

The implementation of the Capacity property is conditional.

Condition: The Capacity property shall be implemented if the representation of the aggregation of host resources is implemented (see 7.4).

If the Capacity property is implemented, its value shall reflect the maximum amount of processing power that can be allocated from the resource pool, in units as expressed by the value of the AllocationUnits property (see 7.3.7). If the CIM_ResourcePool instance represents a processor resource pool with unlimited capacity, the value of the Capacity property shall be set to the largest value supported by the uint64 datatype.

The special value NULL shall be used if the implementation does not have knowledge about the resource capacity represented by the pool. This may reflect a permanent or a temporary situation.

7.3.7 AllocationUnits property

The value of the AllocationUnits property shall be expressed through DSP0004 programmatic units using one these base units: "hertz", "percent" or "count" .

NOTE The units defined by value of the AllocationUnits property applies to the values of the Reserved and the Limit property; it does not apply to the value of the VirtualQuantity property.

7.3.8 Instance requirements

Each processor resource pool shall be represented by a CIM_ResourcePool instance; the provisions of 10.12 and 7.3 apply. The CIM_ResourcePool instance shall be associated with the CIM_System instance representing the host system through an instance of the CIM_HostedResourcePool association (see DSP1041.

7.4 Aggregation of host resources

The implementation of the representation of the aggregation of host processor(s) into a processor resource pool is optional.

If the representation of the aggregation of host processors is implemented, it may be supported for all or for only for some processor resource pools. If the aggregation of host processors is supported for a particular resource pool, any instance of the CIM_Processor class representing a host processor that contributes processing resources into that resource pool shall be associated to the CIM_ResourcePool instance representing the resource pool through an instance of the subclass of CIM_Component association; the provisions of 10.1 apply.

7.5 Processor resource pool hierarchies

The implementation of the representation of processor resource pool hierarchies is optional. If implemented, any concrete processor resource pool shall be represented through a CIM_ResourcePool instance, 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 CIM_ResourcePool instance that represents its parent processor resource pool.
- The instance shall be associated through an instance of the CIM_ElementSettingData association to the RASD instance that represents the amount of processing power allocated from the parent resource pool.

7.6 Default processor resource pool

The implementation of designating a default processor resource pool is optional. If implemented, all of the following conditions apply:

- The default processor resource pool shall be represented by a CIM_ResourcePool instance; see 7.8.3.2
- That instance shall be associated to the CIM_AllocationCapabilities instance that represents the pools default allocation capabilities as specified in 7.8.4.3 .
- The same CIM_AllocationCapabilities instance shall also represent the systems default allocation capabilities as specified in 7.8.3.2.

7.7 Processor resource pool management

The implementation of processor resource pool management is optional.

If implemented, the specifications of DSP1041, clause 7.4 "Resource Pool Management" apply; this profile does not specify specializations or extensions of resource pool management beyond those defined by DSP1041 .

7.8 Processor resource allocation

This subclause details requirements for the representation of resource allocation information through CIM_ResourceAllocationSettingData (RASD) instances.

7.8.1 General

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

Implementations of this profile shall implement the *virtual resource allocation* pattern as defined in DSP1041, 7.2.

7.8.2 Flavors of allocation data

Various flavors of allocation data are defined:

- Processor resource allocation requests; see 6.4.1 .
- Processor resource allocations; see 6.4.2 .
- Settings that define the capabilities or mutability of managed resources. DSP1043 specifies a capabilities model that conveys information about the capabilities and the mutability of managed resources in terms of RASD instances.

- Parameters in operations that define or modify any of the representations listed above. DSP1042 that specifies methods for the definition and modification of virtual resources. These methods use RASD instances for the parameterization of resource-allocation-specific properties.

Table 2 lists acronyms that are used in subclauses of 7.8 in order to designate RASD instances that represent various flavors of processor resource allocation data.

Table 2 – Acronyms for RASD adapted for the representation of various flavors of allocation data

Acronym	Flavor
Q_RASD	RASD adapted for the representation of processor resource allocation requests
R_RASD	RASD adapted for the representation of processor resource allocations
C_RASD	RASD adapted for the representation of settings that define capabilities of systems or processor resource pools, or that define the mutability of processor resource allocations or processor resource allocation requests
D_RASD	RASD adapted for the representation of new processor resource allocation requests in method parameter values
M_RASD	RASD adapted for the representation of modified processor resource allocations or processor resource allocation request in method parameter values

Subclauses of 7.8 detail implementation requirements for property values in RASD instances. In some cases requirements only apply to a subset of the flavors listed in Table 2; this is marked in the text through the use of respective acronyms.

7.8.3 CIM_ResourceAllocationSettingData class

This subclause defines rules for the values of properties in instances of the CIM_ResourceAllocationSettingData (RASD) class representing processor resource allocation information representing the various flavors of processor resource allocation information defined in Table 2.

7.8.3.1 ResourceType property

The value of the ResourceType property in RASD instances representing processor resource allocation information shall be set to 3 (Processor) for processor resource allocation data.

7.8.3.2 PoolID property

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

7.8.3.3 ConsumerVisibility property

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

- A value of 2 (Passed-Through) shall denote that the virtual processor is based on a passed-through host processor .
- A value of 3 (Virtualized) shall denote that processor is virtualized.
- In the cases of { Q_RASD | D_RASD | M_RASD), a value of 0 (Unknown) shall indicate that the processor resource allocation request does not predefine which type of processor shall be allocated.

Other values shall not be used.

7.8.3.4 HostResource[ ] array property

The implementation of the HostResource[ ] array property is conditional.

Condition: The value 2 (Passed-Through) is supported for the value of the ConsumerVisibility property, or any of the values 3 (Dedicated), 4 (Soft Affinity) or 5 (Hard Affinity) is supported for the MappingBehavior property.

If HostResource[ ] array property is implemented, the provisions in this subclause apply.

In the cases of { Q_RASD | C_RASD | D_RASD | M_RASD } the value of the HostResource[ ] array property shall refer to (the representation of) one or more host resources that are configured to contribute processing power for the processor resource allocation.

In the case of R_RASD the value of the HostResource[ ] array property shall refer to (the representation of) the host resource(s) that contribute processing power for the processor resource allocation.

Elements of the value of the HostResource[ ] array property shall refer to instances of CIM classes, using the WBEM URI format as specified by DSP0207.

7.8.3.5 AllocationUnits property

The value of the AllocationUnits property shall be expressed in one of the following programmatic units: "hertz", "percent" or "count" or a multiple of the units expressed through a regular expression, as defined in DSP0004 programmatic units.

NOTE The units defined by value of the AllocationUnits property applies to the values of the Reserved and the Limit property; it does not apply to the value of the VirtualQuantity property.

7.8.3.6 VirtualQuantity property

The value of the VirtualQuantity property shall denote the number of virtual processors available to a virtual system.

NOTE The value of the VirtualQuantity property is a count; it is not expressed in allocation units. The units used for VirtualQuantity may be expressed in VirtualQuantityUnits (see 7.8.3.11).

7.8.3.7 Reservation property

The implementation of the Reservation property is optional.

If the Reservation property is implemented, the value of the Reservation property shall denote the minimum amount of host processing resources reserved for the use of a virtual system, expressed in allocation units.

7.8.3.8 Limit property

The implementation of the Limit property is optional.

If the Limit property is implemented, the value of the Limit property shall denote the maximum amount (or limit) of host processing power available to a virtual system, expressed in allocation units.

7.8.3.9 Weight property

The implementation of the Weight property is optional.

If the Weight property is implemented, its value shall denote the relative priority of a processor resource allocation in relation to other processor resource allocations from the same resource pool.

7.8.3.10 MappingBehavior property

The implementation of the MappingBehavior property is optional.

If the MappingBehavior property is implemented, its value shall denote how host resources referenced by elements in the value of HostResource[ ] array property relate to the processor resource allocation.

In R_RASD instances the following rules apply to the value of the MappingBehavior property:

- A value of 2 (Dedicated) shall indicate that the represented processor resource allocation is provided by host processor 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 processor resource allocation is provided using host processor resource as referenced by the value of the HostResource[ ] array property.
- Other values shall not be used.

In Q_RASD instances the following rules apply to the value of the MappingBehavior property:

- The special value NULL or a value of 0 (Unknown) shall indicate that the processor resource allocation request does not require specific host resources.
- A value of 2 (Dedicated) shall indicate that the processor resource allocation request shall be provided by exclusively dedicated host processor resources as specified through the value of the HostResource[ ] array property.
- A value of 3 (Soft Affinity) shall indicate that the processor resource allocation request shall preferably be provided by host processor 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 processor resource allocation request shall preferably be provided by host processor 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.

7.8.3.11 VirtualQuantityUnits property

VirtualQuantityUnits is an optional property that may be used to denote the units of the virtual device being allocated from the resource pool. The value of VirtualQuantityUnits shall be "count".

7.8.3.12 ResourceSubType property

The value of the ResourceSubType property shall designate a resource subtype. The format of the value shall be as follows: "<org-id>:<org-specific>". The <org-id> part shall identify the organization that defined the resource subtype value; the <org-specific> part shall uniquely identify a resource subtype within the set of subtype defined by the respective organization.

7.8.4 Instance requirements

This subclause details processor resource allocation related instance requirements.

7.8.4.1 Representation of resource allocation requests

Each processor resource allocation request shall be represented by a Q_RASD instance; the provisions of 10.11 apply.

7.8.4.2 Representation of resource allocations

Each processor resource allocation shall be represented by a R_RASD instance; the provisions of 10.11 apply.

The R_RASD instance shall be associated to the Q_RASD instance representing the corresponding resource allocation request (see 7.8.4.1) through an instance of the CIM_ElementSettingData association; the provisions of 10.10 apply.

The R_RASD instance shall be associated to the CIM_ResourcePool instance providing resources for the allocation (see 7.3.8) through an instance of the CIM_ResourceAllocationFromPool association; the provisions of 10.4 apply.

Implementations may represent a resource allocation request and the corresponding resource allocation by one RASD instance; in this case the association requirements of this subclause apply correspondingly. Note that association instances that refer to the RA_SASD instance are only existent while the resource is allocated.

7.8.4.3 Representation of resource allocation capabilities

The allocation capabilities of a system or a resource pool shall be represented by a CIM_AllocationCapabilities instance that is associated to the CIM_System instance representing the system or to the CIM_ResourcePool instance representing the resource pool through an instance of the CIM_ElementCapabilities association; see DSP1043.

The settings that define the allocation capabilities of a processor resource pool shall be represented by C_RASD instances; the provisions of 10.11 apply.

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

7.8.4.4 Representation of resource allocation mutability

The mutability of a resource allocation or resource allocation request shall be represented by a CIM_AllocationCapabilities instance that is associated to the RASD instance representing the resource allocation or resource allocation request through an instance of the CIM_ElementCapabilities association; see DSP1043.

The settings that define the allocation capabilities of a processor resource pool shall be represented by C_RASD instances; the provisions of 10.11 apply.

7.9 Virtual processor

A virtual processor shall be represented by exactly one CIM_Processor instance; the provisions of 10.8 apply. That instance shall be associated to all of the following instances:

- the CIM_ComputerSystem instance that represents the virtual system through an instance of the CIM_SystemDevice association; the provisions of 10.15 apply.
- the RASD instance that represents processor resource allocation through an instance of the CIM_SettingsDefineState association; the provisions of 10.13 apply.
- the CIM_ResourcePool instance that represents the processor resource pool providing the resource allocation through an instance of the CIM_ElementAllocatedFromPool association; the provisions of 10.2 apply.

Implementations may implement DSP1022 for virtual processors.

8 Methods

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

8.1 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 DSP1041 and DSP1043. 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.

8.2 CIM_Processor for host processors

All operations in the default list in 8.1 shall be implemented as defined in DSP0200. Note that related profiles may define additional requirements on operations for the profile class.

8.3 CIM_Processor for virtual processor

All operations in the default list in 8.1 shall be implemented as defined in DSP0200. Note that related profiles may define additional requirements on operations for the profile class.

8.4 CIM_ReferencedProfile

All operations in the default list in 8.1 shall be implemented as defined in DSP0200. Note that related profiles may define additional requirements on operations for the profile class.

8.5 CIM_RegisteredProfile

All operations in the default list in 8.1 shall be implemented as defined in DSP0200. Note that related profiles may define additional requirements on operations for the profile class.

8.6 CIM_SystemDevice for host processors

All operations in the default list in 8.1 shall be implemented as defined in DSP0200. Note that related profiles may define additional requirements on operations for the profile class.

8.7 CIM_SystemDevice for virtual processors

All operations in the default list in 8.1 shall be implemented as defined in DSP0200. Note that related profiles may define additional requirements on operations for the profile class.

9 Use cases

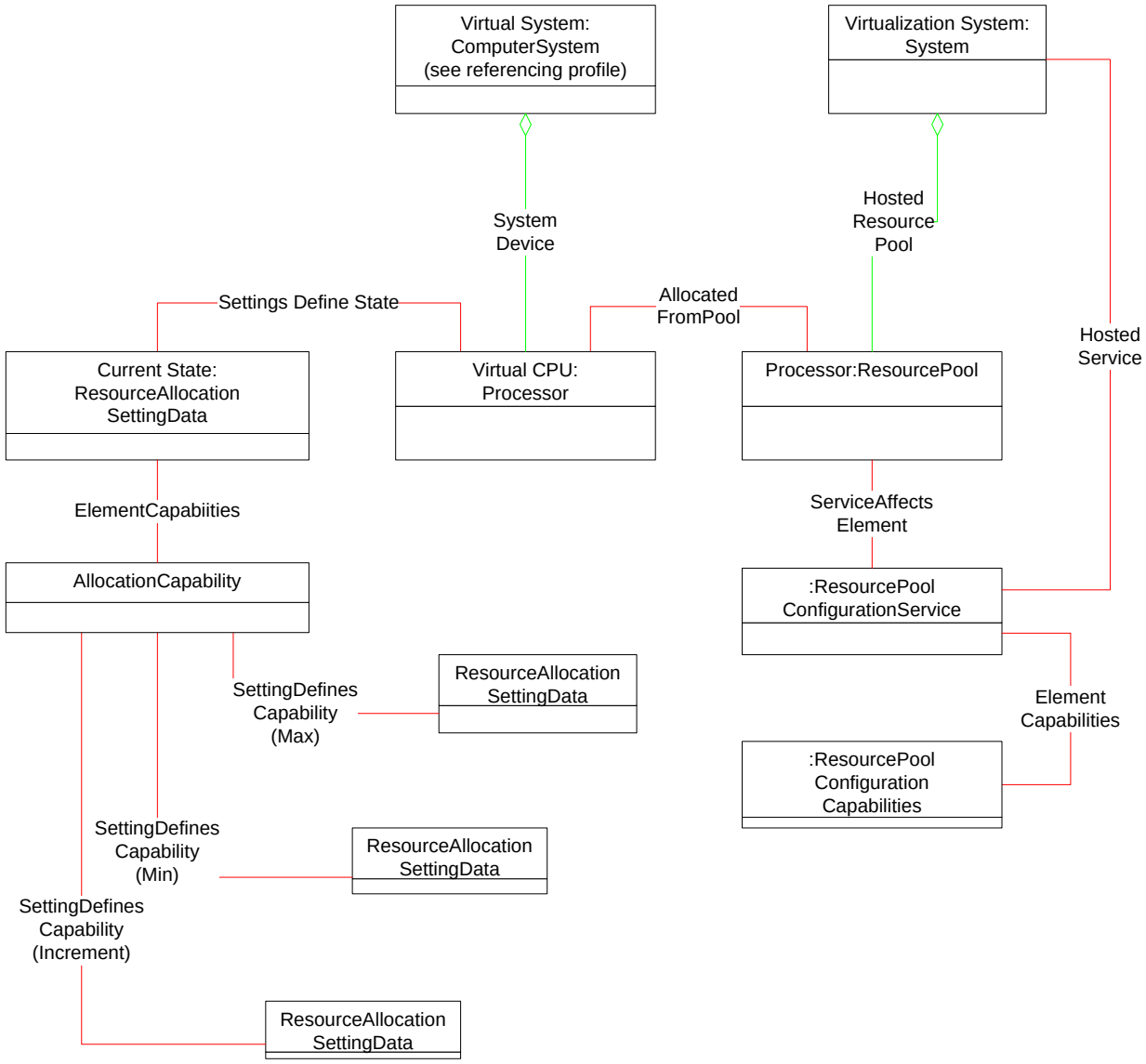
Clause 9 of DSP1042 describes use cases virtual system definition, modification and destruction.

Clause 9 of DSP1057 describes use cases for discovery of virtual systems, determination of the state and properties of a virtual system and the defined virtual system. This clause is a pre-requisite to understanding the following use cases.

Clause 9 of DSP1059 covers a number of essential use cases and should be read.

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.

Figure 2 is a general instance diagram showing the essential classes for processor resource allocation.



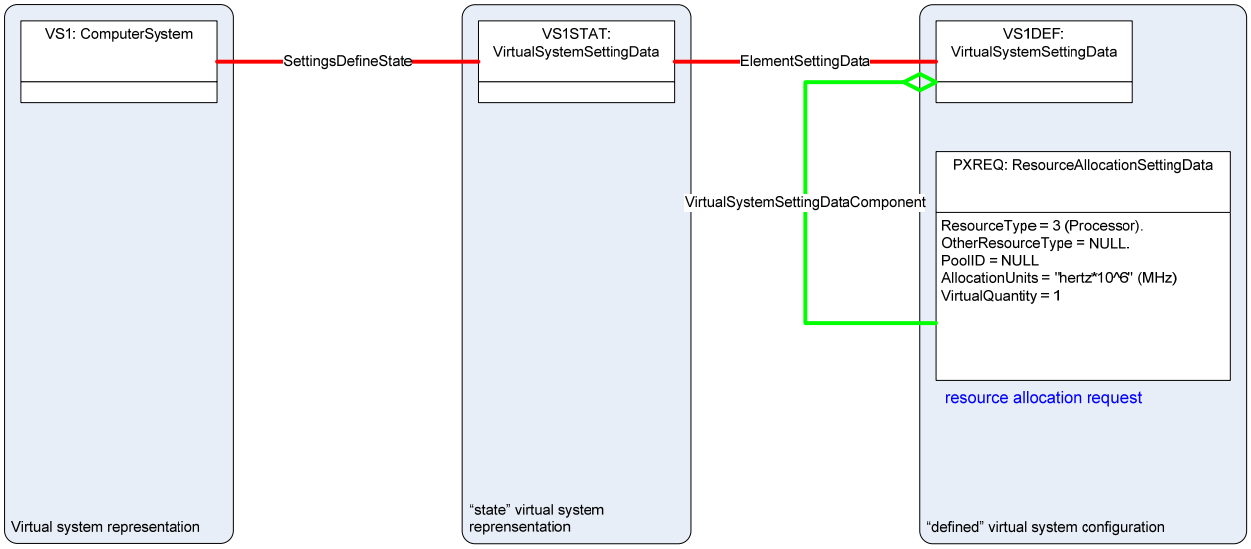
796

797 **Figure 2 – Processor Resource Virtualization Profile: Instance diagram**

798 There are different allocation units that may be used for the processor resource pool.

799 If instances of CIM_processor are instantiated to represent the physical processors aggregated into the
800 primordial pool those instances should conform to DSP1022.

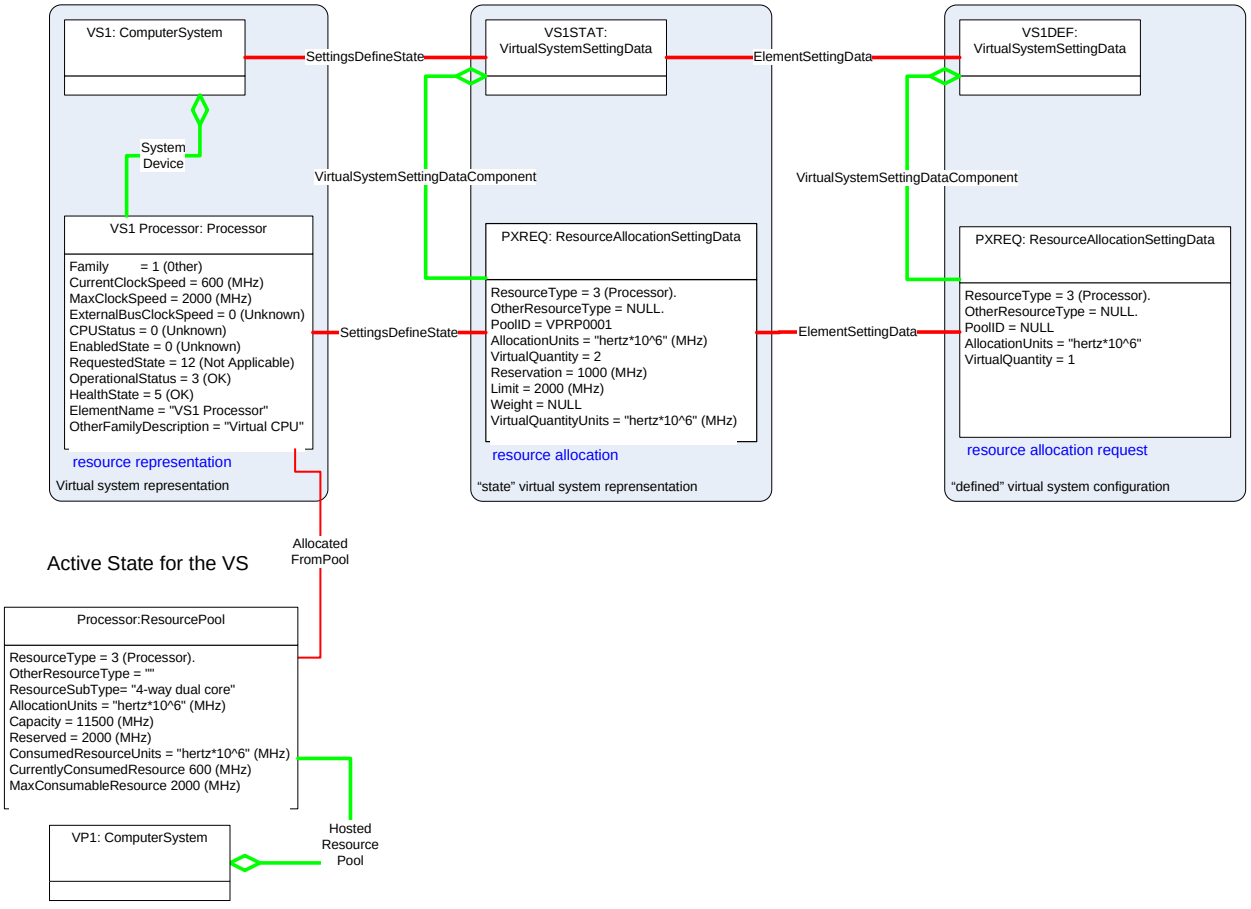
801 A defined state for allocation of processor resources is shown in Figure 3.



802

803 **Figure 3 – Defined state**

804 An active state for allocation of processor resources is shown in Figure 4.



805

806 **Figure 4 – Active state**

9.1 Use case 1 – Increase the allocated processor capacity to a virtual machine in MHz.

Pre-conditions:

The virtual system is active.

The CIM_ComputerSystem instance representing the virtual system is known.

The CIM_ResourcePool instance representing the processor resource pool is known.

The CIM_AllocationUnits of the resource pool is “hertz*10^6”.

The defined CIM_ResourceAllocationSettingData for the processor has property Reservation = 1000 (i.e 1GHz of processer capacity is allocated to the existing active virtual system) and has property Limit = 3000. The Reservation represents the minimum guaranteed resource available, even when system overcommitted. The sum of the Reservations must be less than the capacity of the resource pool. The Limit represents the maximum resource consumption, even when under committed.

Main:

The client changes the defined CIM_ResourceAllocationSettingData for the processor property to Reservation = 2000.

The client invokes the CIM_VirtualManagementService.ModifyResourceSettings() method.

Check the return code value for success execution.

Post-conditions:

The minimum processor capacity required to be allocated for the support of the virtual processor is now 2 GHz, the maximum admissible processor capacity is now 3 GHz.

9.2 Use case 2 – Increase the allocated processor capacity to a virtual machine in percent.

Pre-conditions:

The virtual system is active.

The CIM_ComputerSystem instance representing the virtual system is known.

The CIM_ResourcePool instance representing the processor resource pool is known.

The CIM_ResourceAllocationSettingData.AllocationUnits of the resource pool is “%”.

The defined CIM_ResourceAllocationSettingData for the processor has property Reservation = 10 (i.e ten percent of processer capacity is allocated to the existing active virtual system) and has property Limit = 15.

Main:

The client changes the defined CIM_ResourceAllocationSettingData for the processor property Reservation = 15.

The client invokes the CIM_VirtualSystemManagementService.ModifyResourceSettings() method.

Check the return code value for success execution.

Post-conditions:

The processor resource allocated to the virtual system is now 15 percent minimum and the maximum processor resource that the virtual system can consume is 15 percent.

9.3 Use case 3 - Add a virtual processor to a virtual system

The virtual system is in-active.

The CIM_ComputerSystem instance representing the virtual system is known.

The CIM_ResourcePool instance representing the processor resource pool is known.

TheCIM_ResourceAllocationSettingData.AllocationUnits of the resource pool is “hertz*10^6”.

Figure 5 illustrates the instance diagram before the CIM_VirtualSystemManagementService.ModifyResourceSettings() method is called.

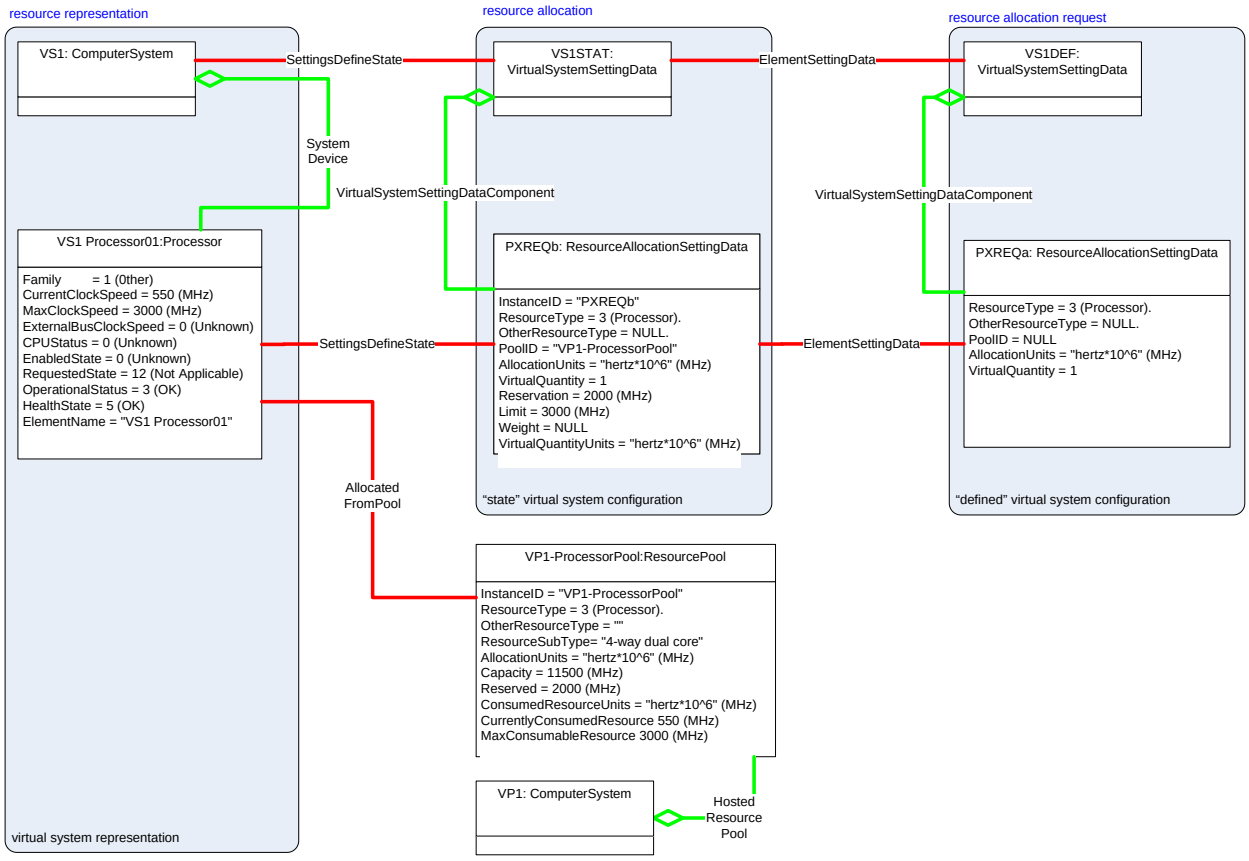
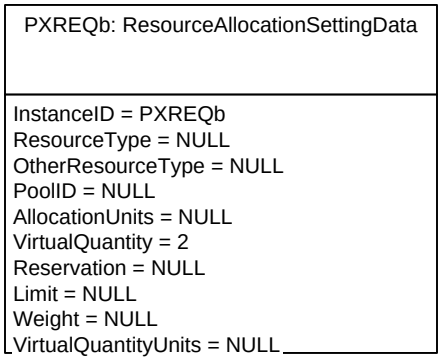


Figure 5 – CIM_ModifyResourceSettings – Before

Figure 6 illustrates the resource allocation setting data transferred with the CIM_VirtualSystemManagementService.ModifyResourceSettings() method call to increase the number of virtual processors.



857

858 **Figure 6 – RASD to Modify Resources**

859 Main:

860 Check the maximum value of the CIM_ResourceAllocationSettingData.VirtualQuantity to verify that a
861 virtual processor addition is allowed.

862 Check the increment value of the CIM_ResourceAllocationSettingData.VirtualQuantity to verify that a one
863 processor addition is allowed.

864 The client changes the CIM_ResourceAllocationSettingData.VirtualQuantity by one.

865 The client invokes the CIM_VirtualSystemManagement.Service.ModifyResourceSettings() method.

866 Check the return code value for success execution.

867 Post-conditions:

868 The virtual quantity of processors allocated to the virtual system is now one greater.

869 Figure 7 illustrates the instance diagram after the
870 CIM_VirtualSystemManagementService.ModifyResourceSettings() method is called.

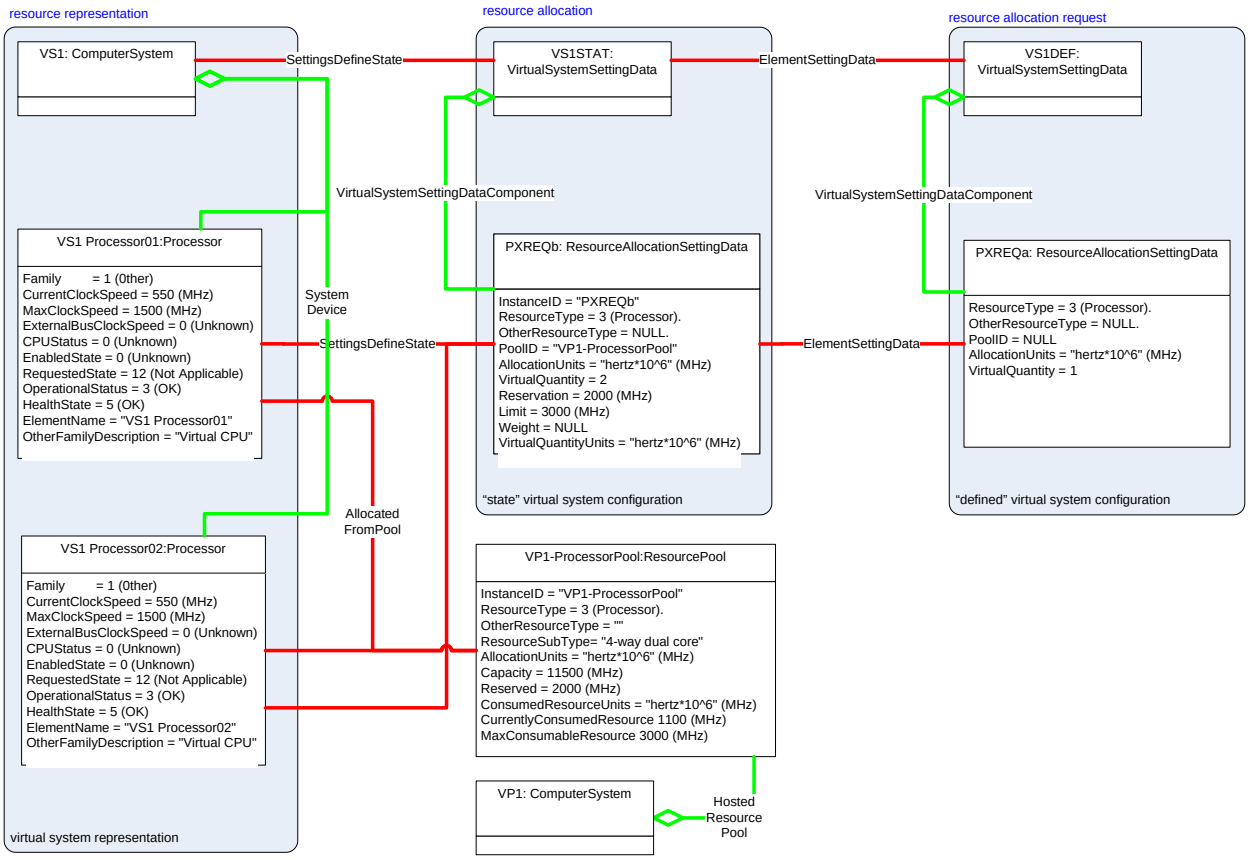


Figure 7 – CIM_ModifyResourceSettings - After

9.4 Use case 4 – Allocation of processor resource by weight

Pre-conditions:

The virtual system is in-active.

The CIM_ComputerSystem instance representing the virtual system is known.

The CIM_ResourcePool instance representing the processor resource pool is known.

The CIM_AllocationUnits of the resource pool is “hertz*10^6”.

The defined CIM_ResourceAllocationSettingData for the processor has property Reservation = 0 (i.e no minimum level of processer capacity is allocated to the existing active virtual system) and has property Limit = none.

Main:

The client changes the defined CIM_ResourceAllocationSettingData for the processor property Weight = 200.

The client invokes the CIM_VirtualSystemManagementService.ModifyResourceSettings() method.

Check the return code value for success execution.

The client activates the virtual system.

888 Post-conditions:

889 The processor resource allocated to the virtual system is now 200/ SUM (Weight for each active virtual
890 system). For example, if there are three other virtual systems with their Weight properties set to 200, 300,
891 100, then this virtual system is allocated 200/(200+300+100+200) or 25% of the processer capacity.

892 9.5 Use case 5 – Allocation of an additional processor resource

893 Pre-conditions:

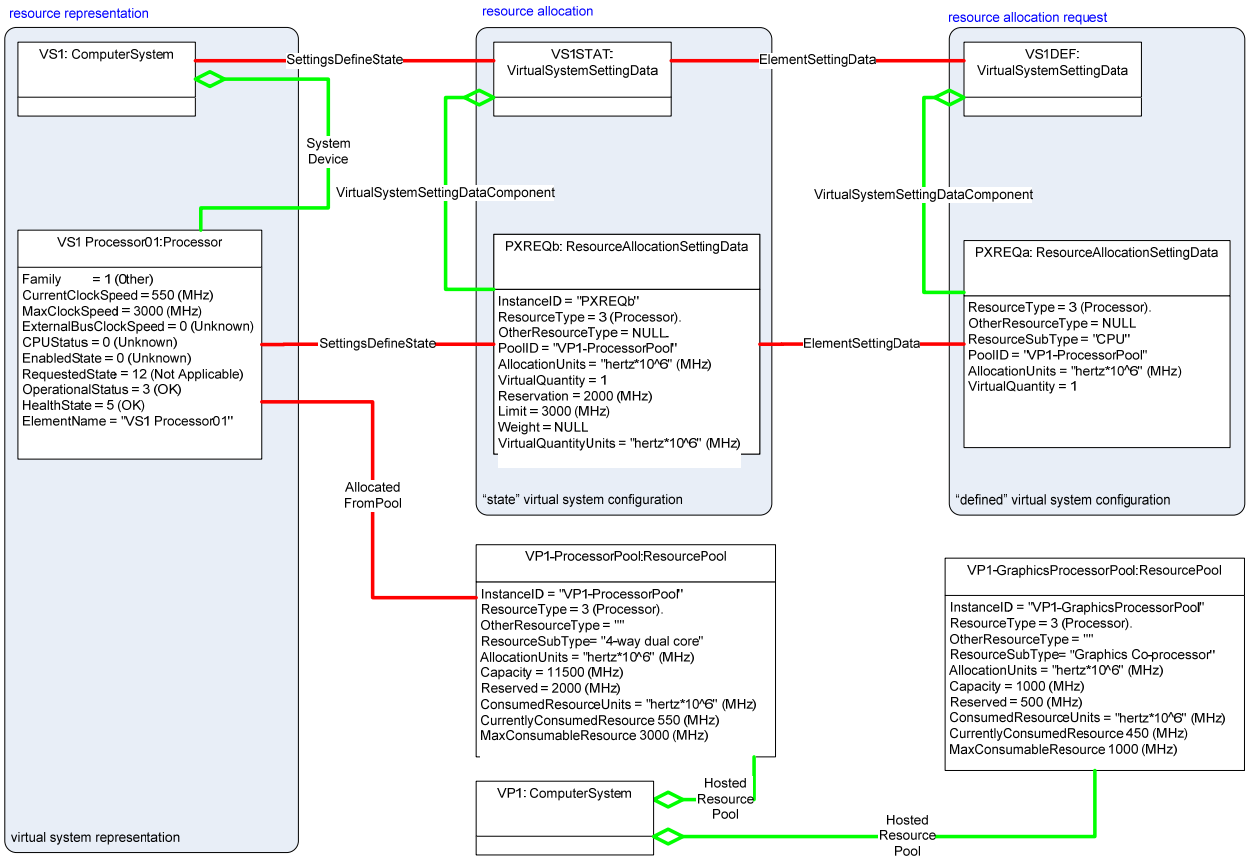
894 The virtual system is active.

895 The CIM_ComputerSystem instance representing the virtual system is known.

896 The CIM_ResourcePool instances representing the processor resource pools is known.

897 The CIM_AllocationUnits of the resource pools is “hertz 10^6”.

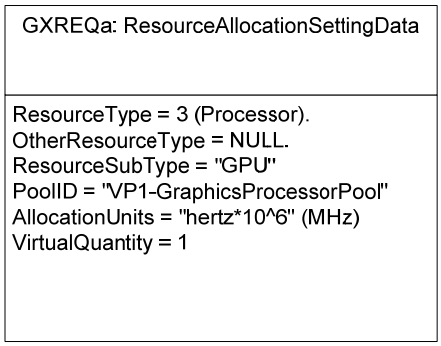
898 Figure 7 illustrates the instance diagram before the
899 CIM_VirtualSystemManagementService.AddResourceSettings() method is called.



900

901 **Figure 8 – CIM_AddResourceSettings - Before**

902 Figure 9 illustrates the resource allocation setting data transferred with the
903 CIM_VirtualSystemManagementService.AddResourceSettings() method call to add an additional virtual
904 processor to the virtual system.



905

906 **Figure 9 – RASD to add processor**

907 Main:

908 The client creates a defined CIM_ResourceAllocationSettingData specifying the PoolID for the resource
909 pool from which the additional processor is to be allocated.

910 The client invokes the CIM_VirtualSystemManagementService.AddResourceSettings() method.

911 Check the return code value for success execution.

912 Post-conditions:

913 The processor resources allocated to the virtual system is now two processors, one from the “VP1-
914 ProcesssorPool” and one from the“VP1-GraphicsProcesssorPool”. Figure 10 illustrates the instance
915 diagram after the CIM_VirtualSystemManagementService.AddResourceSettings() method is called.

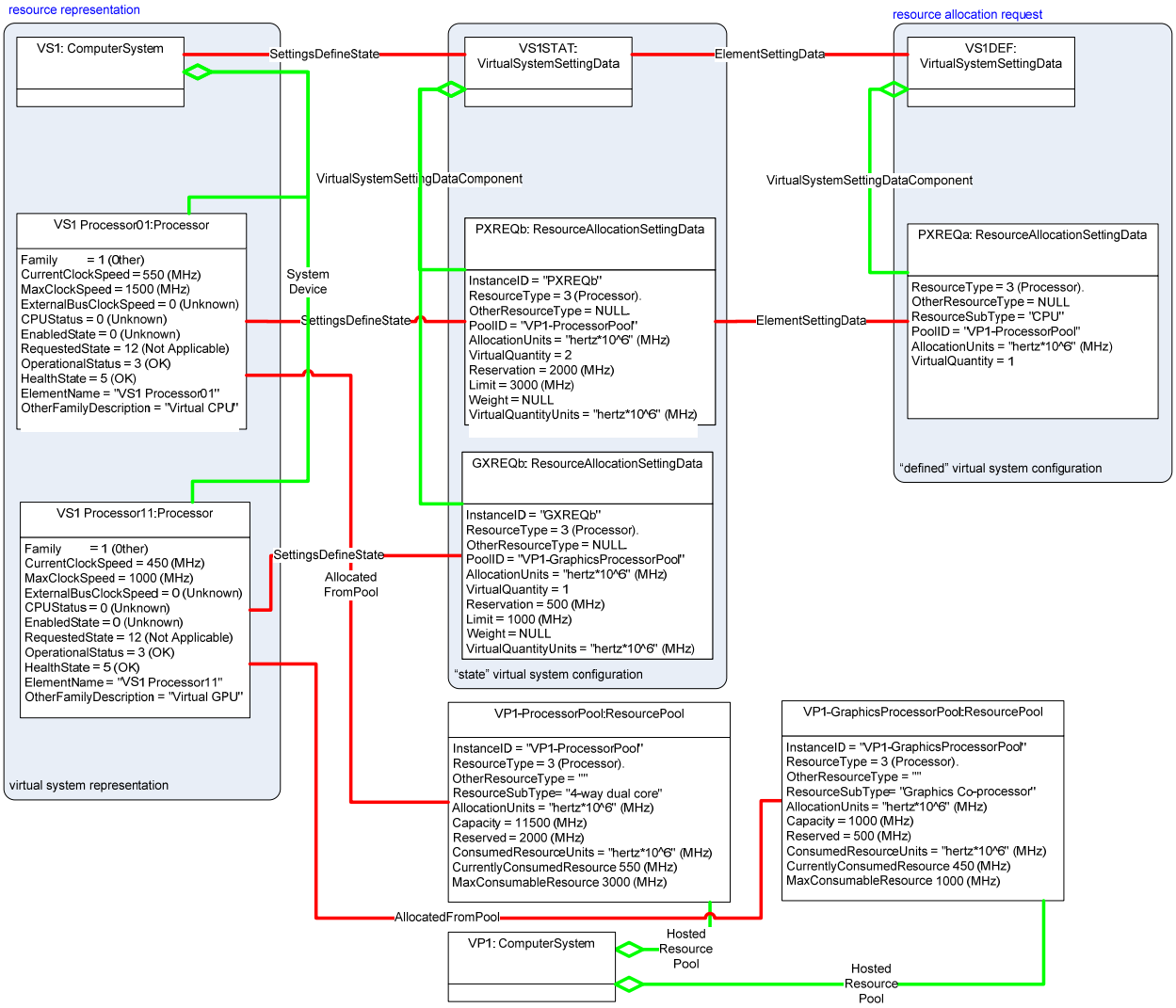


Figure 10 – CIM_AddResourceSettings - After

10 CIM elements

Table 3 lists CIM elements that are adapted by this profile. Each CIM element shall be implemented as described in Table 3. The CIM Schema descriptions for any referenced element and its sub-elements apply.

Clauses 7 ("Implementation") and 8 ("Methods") may impose additional requirements on these elements.

Table 3 – CIM Elements: Processor Resource Virtualization Profile

Element Name	Requirement	Description
CIM_AffectedJobElement	Optional	See DSP1041 .
CIM_AllocationCapabilities for capabilities	Mandatory	See DSP1043 .
CIM_AllocationCapabilities for mutability	Optional	See DSP1043 .
CIM_Component for resource pool	Conditional	See 10.1.
CIM_ConcreteJob	Optional	See DSP1041 .
CIM_ElementAllocatedFromPool for allocated virtual processors	Mandatory	See 10.2 .
CIM_ElementAllocatedFromPool for resource pool hierarchies	Conditional	See 10.3 .
CIM_ElementCapabilities for capabilities	Mandatory	See DSP1043 .
CIM_ElementCapabilities for mutability	Conditional	See DSP1043 .
CIM_ElementCapabilities for resource pools	Mandatory	See DSP1041 .
CIM_ElementSettingData for processor resource allocation	Mandatory	See 10.4 .
CIM_ElementSettingData for processor resource pools	Conditional	See 10.5 .
CIM_HostedDependency	Optional	See 10.6 .
CIM_HostedResourcePool	Mandatory	See DSP1041 .
CIM_HostedService	Mandatory	See DSP1041 .
CIM_Processor for host processors	Conditional	See 10.7 .
CIM_Processor for virtual processors	Mandatory	See 10.8 .
CIM_ReferencedProfile	Mandatory	See DSP1057 .
CIM_RegisteredProfile	Mandatory	See 10.9 .
CIM_ResourceAllocationFromPool	Optional	See 10.10 .
CIM_ResourceAllocationSettingData	Mandatory	See 10.11 .
CIM_ResourcePool	Mandatory	See 10.12 .
CIM_ResourcePoolConfigurationCapabilities	Mandatory	See DSP1041 .
CIM_ResourcePoolConfigurationService	Mandatory	See DSP1041 .
CIM_SettingsDefineState	Mandatory	See 10.13 .
CIM_ServiceAffectsElement	Mandatory	See DSP1041 .
CIM_SystemDevice for host processors	Conditional	See 10.14 .
CIM_SystemDevice for virtual processors	Mandatory	See 10.15 .
Indications		
None defined		

10.1 CIM_Component for resource pool

The implementation of the CIM_Component association for the representation of the aggregation of host resources into resource pools is conditional.

927 Condition: The representation of resource aggregation (see 7.4) is implemented.

928 The CIM_Component association is abstract; therefore it cannot be directly implemented. For this reason
929 the provisions in this subclause shall be applied to implementations of subclasses of the CIM_Component
930 association. However, note that clients may directly resolve abstract associations without knowledge of
931 the concrete subclass that is implemented.

932 Table 4 lists the requirements for elements of this association. These requirements are in addition to
933 those specified in the CIM Schema and in DSP1041 .

934 **Table 4 – Association: CIM_Component for resource pool**

Elements	Requirement	Notes
GroupComponent	Mandatory	**Key:** Value shall reference the CIM_ResourcePool instance that represents the resource pool. **Cardinality:** 0..1
PartComponent	Mandatory	**Key:** Value shall reference the CIM_ManagedElement instance that represents a component of the resource pool. **Cardinality:** *

935 10.2 CIM_ElementAllocatedFromPool for allocated virtual processors

936 Table 5 lists the requirements for elements of this association. These requirements are in addition to
937 those specified in the CIM Schema and in DSP1041.

938 **Table 5 – Association: CIM_ElementAllocatedFromPool**

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

939 10.3 CIM_ElementAllocatedFromPool for resource pool hierarchies

940 The implementation of the CIM_ElementAllocatedFromPool association for the representation of resource
941 pool hierarchies is conditional.

942 Condition: Resource pool management (see 7.7) is implemented.

943 Table 6 lists the requirements for elements of this association. These requirements are in addition to
944 those specified in the CIM Schema and in DSP1041 .

945 **Table 6 – Association: CIM_ElementSettingData**

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** Value shall reference the CIM_ResourcePool instance that represents the parent resource pool. **Cardinality:** 1
Dependent	Mandatory	**Key:** Value shall reference the CIM_ResourcePool instance that represents the child resource pool. **Cardinality:** *

946 ## 10.4 CIM_ElementSettingData for processor resource allocation

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

949 **Table 7 – Association: CIM_ElementSettingData for processor resource allocation**

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** Value shall reference the RASD instance represents the processor resource allocation. **Cardinality:** 1
SettingData	Mandatory	**Key:** Value shall reference the RASD instance that represents the processor resource allocation request. **Cardinality:** 0..1
IsDefault	Mandatory	Value shall be 1 (Is Default).

950 ## 10.5 CIM_ElementSettingData for processor resource pool

951 The implementation of the CIM_ElementSettingData class for the representation of the relationship
952 between a child resource pool and its processor resource allocation is conditional.

953 Condition: Processor resource pool hierarchies (see 7.5) are implemented.

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

956 **Table 8 – Association: CIM_ElementSettingData (Processor Resource Pool)**

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

10.6 CIM_HostedDependency

The support of the CIM_HostedDependency association is optional.

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_HostedDependency

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** Value shall reference the CIM_Processor instance that represents host processor. **Cardinality:** 0..1
Dependent	Mandatory	**Key:** Value shall reference the CIM_Processor instance that represents virtual processor. **Cardinality:** 0..1

10.7 CIM_Processor (host processor)

The implementation of the CIM_Processor class for the representation of host processors is conditional.

Condition: The representation of host resources is implemented; see 7.2.

Table 10 lists the requirements for elements of this class. These requirements are in addition to those specified in the CIM Schema and in DSP1022 if that is implemented.

Table 10 – Class: CIM_Processor (host processor)

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	**Key**
CreationClassName	Mandatory	**Key**
SystemName	Mandatory	**Key**
Name	Mandatory	**Key**
EnabledState	Mandatory	See CIM schema description.
RequestedState	Mandatory	See CIM schema description.

10.8 CIM_Processor (virtual processor)

See 7.9 for detailed implementation requirements for this class adaptation.

Table 11 lists the requirements for elements of this class adaptation. These requirements are in addition to those specified in the CIM Schema and in DSP1022 if that is implemented.

Table 11 – Class: CIM_Processor (virtual system)

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

10.9 CIM_RegisteredProfile

The basic adaptation of the CIM_RegisteredProfile class is specified by DSP1033.

Table 12 lists the requirements for elements of this class. These requirements are in addition to those specified in DSP1033.

Table 12 – Class: CIM_RegisteredProfile

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

10.10 CIM_ResourceAllocationFromPool

The support of the CIM_ResourceAllocationFromPool association is optional.

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

Table 13 – Association: CIM_ResourceAllocationFromPool

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

10.11 CIM_ResourceAllocationSettingData

See 7.8.3 for detailed implementation requirements for this class.

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

987 **Table 14 – Class: CIM_ResourceAllocationSettingData**

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**; see DSP1041.
ResourceType	Mandatory	Value shall be 3 (Processor).
OtherResourceType	Mandatory	Value shall be NULL.
ResourceSubType	Optional	See 7.8.3.12 .
PoolID	Mandatory	See 7.8.3.2.
ConsumerVisibility	Optional	See 7.8.3.3.
HostResource[ ]	Optional	See 7.8.3.4.
AllocationUnits	Mandatory	See 7.8.3.5 .
VirtualQuantity	Mandatory	See 7.8.3.6.
Reservation	Optional	See 7.8.3.7.
Limit	Optional	See 7.8.3.8.
Weight	Optional	See 7.8.3.9.
AutomaticAllocation	Optional	See DSP1041.
AutomaticDeallocation	Optional	See DSP1041.
MappingBehavior	Optional	See 7.8.3.10.
VirtualQuantityUnits	Optional	See 7.8.3.11

988 ## 10.12 CIM_ResourcePool

989 Table 15 lists the requirements for elements of this class. These requirements are in addition to those
990 specified in the CIM Schema and in DSP1041.

991 **Table 15 – Class: CIM_ResourcePool**

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ElementName	Optional	See DSP1041.
PoolID	Mandatory	See 7.3.3.
Primordial	Mandatory	See 7.3.4.
Capacity	Conditional	See 7.3.6.
Reserved	Optional	See 7.3.5.
ResourceType	Mandatory	Value shall be 3 (Processor).
OtherResourceType	Mandatory	Value shall be NULL.
ResourceSubType	Optional	See 7.3.2.
AllocationUnits	Mandatory	See 7.1

10.13 CIM_SettingsDefineState

Table 16 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema and in DSP1041.

Table 16 – Association: CIM_SettingsDefineState

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

10.14 CIM_SystemDevice for host processor

The implementation of the CIM_SystemDevice association for the representation of the relationship between host processors and their system is conditional.

Condition: The representation of host resources is implemented; see 7.2.

NOTE If DSP1022 is implemented for host processors, the implementation of the CIM_SystemDevice association for the representation of the relationship between host processors and their system is required.

If the CIM_SystemDevice association is implemented for the representation of the relationship between host processors and their systems, the provisions in this subclause apply.

Table 17 lists the requirements for elements of this association. These requirements are in addition to those specified in the CIM Schema, in DSP1041, and in DSP1022 if that is implemented.

Table 17 – Association: CIM_SystemDevice (Host Processor)

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_Processor class. **Cardinality:** *

10.15 CIM_SystemDevice for virtual processor

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_SystemDevice (Virtual Processor)

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_Processor class. **Cardinality:** *

ANNEX A
(Informative)

Change Log

Version	Date	Description
1.0.0a	2009-09-14	Release as work in progress
1.0.0c	2010-02-02	Release as work in progress
1.0.0	2010-04-22	Release as DMTF Standard