



December 3-6, 2007, Santa Clara Marriott, Santa Clara, CA

CIM System Virtualization Model

Ron Goering

Co-Chair DMTF System Virtualization
Partitioning and Clustering Workgroup

Distinguished Engineer

IBM



Disclaimer

- The information in this presentation represents a snapshot of work in progress within the DMTF.
- This information is subject to change. The Standard Specifications remain the normative reference for all information.
- For additional information, see the Distributed Management Task Force (DMTF) Web site.
- <http://www.dmtf.org/standards/smash>

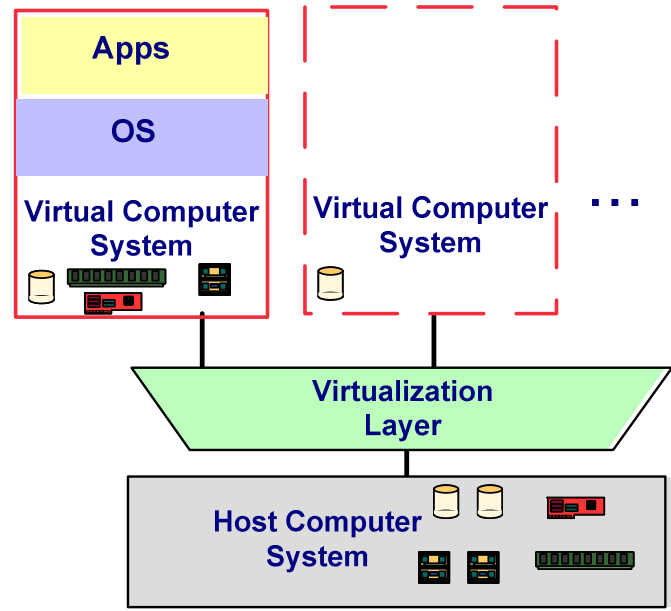
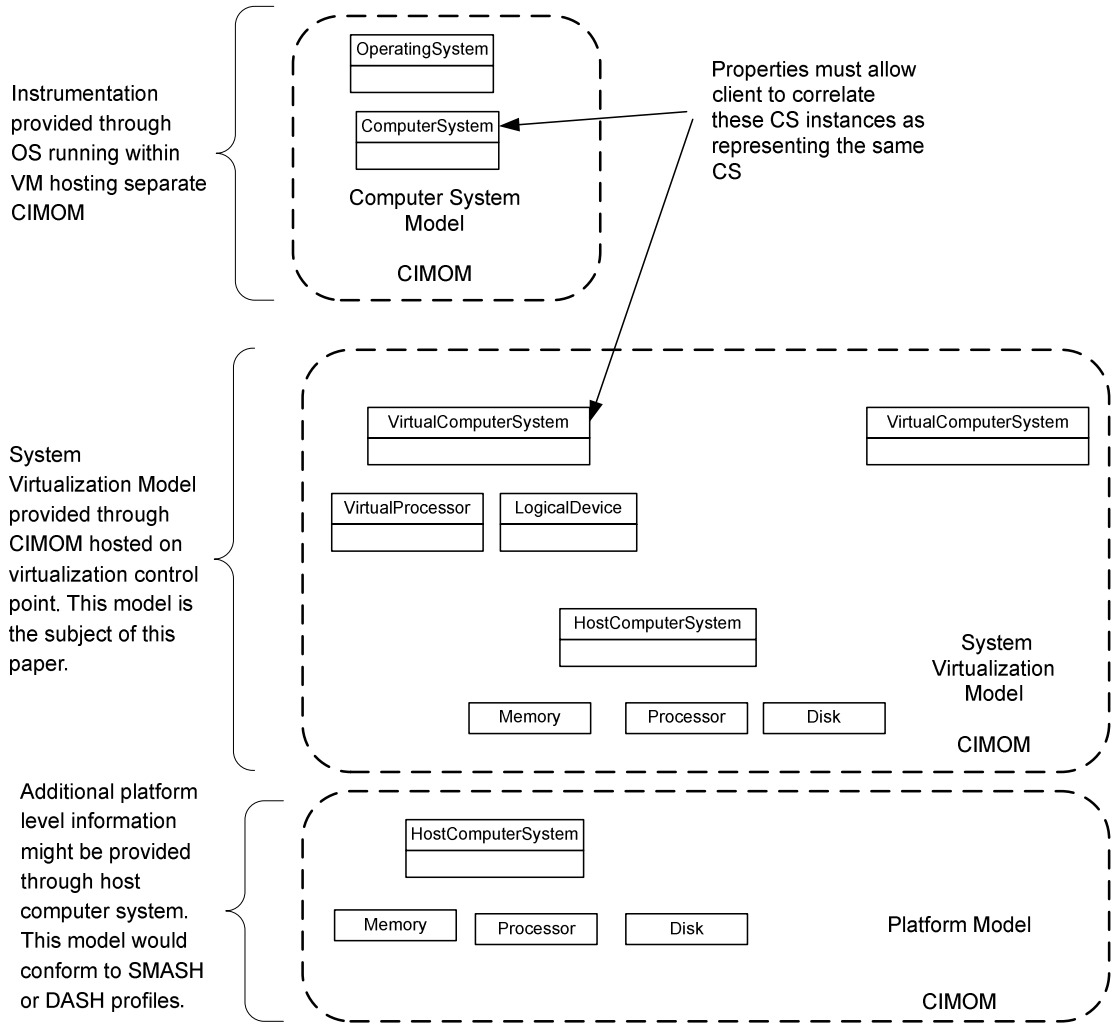


The DMTF was formed to lead the development, adoption and unification of management standards and initiatives for desktop, enterprise and internet environments

System Virtualization WG

- System Virtualization Partitioning and Clustering (SVPC) DMTF Workgroup with participation from VMware, Microsoft, IBM, HP, Sun, Novell, XenSource, Hitachi, Intel, Dell and others
 - Weekly calls (Thurs 9 PDT – details on website:
<http://www.dmtf.org/apps/org/workgroup/redundancy/>)
- Work ongoing to produce CIM model (CIM profile and associated CIM schema changes) for virtual systems and the virtual resources which compose them.
 - Leverage SMI-S profiles for storage virtualization
- Deliverables:
 - System Virtualization Model White Paper
 - http://www.dmtf.org/standards/published_documents/DSP2013_1.0.0.pdf
 - CIM schema request for changes with associated MOF –
 - Changes in CIM 2.15, 2.16, small updates in 2.17
 - CIM Profiles published as draft standards
 - Resource Allocation, Resource Capabilities abstract profiles
 - System Virtualization, Virtual System Autonomous Profiles
 - “Device” profiles: Generic Device,
 - CIM Profiles work in progress
 - Processor, Memory, Block back disks, file back disks, Virtual enet, VHBA, Removable Media,
 - Others in second phase

Virtualization Modeling Environment



Virtual System Model Requirements

- **General Requirements:**

- Enable management applications which are unaware of virtualization to manage virtual systems, i.e. once a ComputerSystem is created most **management operations** (list, install, configure) are **enabled without requiring the management application to understand virtualization**.
 - Make sure appropriate profiles from SMWG, SMI-S and others are applicable
- Support the **symmetry** inherent in multiple layers of virtualization .
- Model should be general and flexible enough to support known virtualization systems including partitioning and containers

- **ComputerSystem**

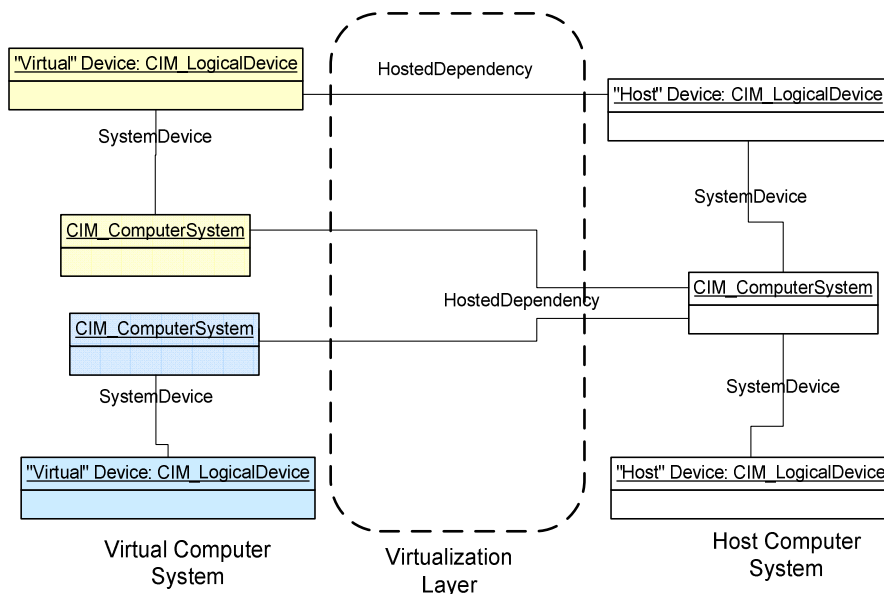
- **Enumerate** virtual systems, resources, relationships on a particular platform.
- **Create** Virtual System specifying resources (CPU, disk, I/O) and attributes about those resources (shared, virtualized, based on what platform resource)
 - Provide appropriate defaults wherever possible
 - Provide ability for management application to introspect the system at runtime to find out virtualization capabilities and resources supported.
- **Delete** virtual system and return resources to platform.
- **Modify** the resources that compose virtual system.

- **Virtual Resources**

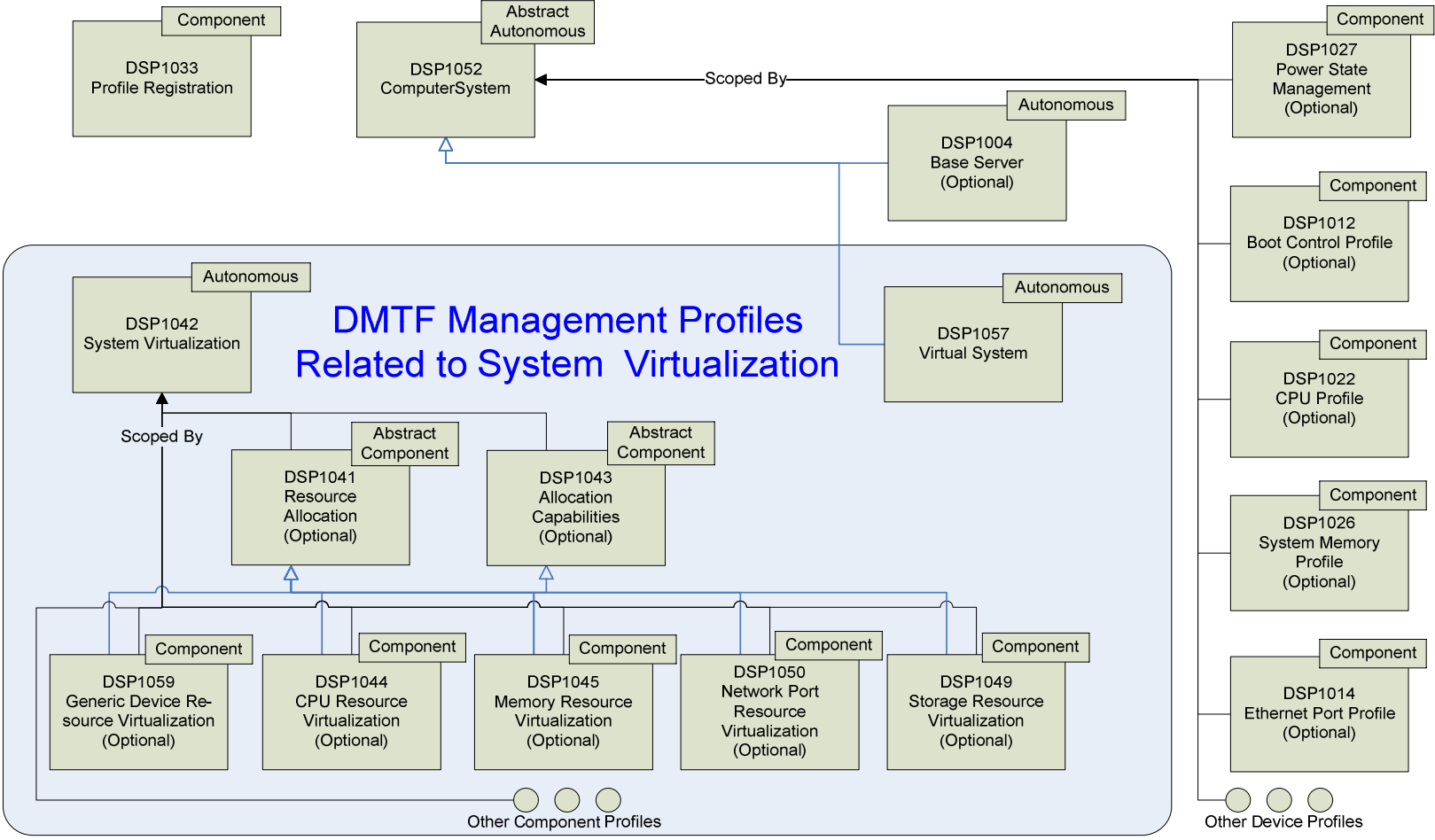
- Support **creation, modification, deletion and inventory of virtual resources**
- Enable **mapping of virtual resources** to underlying resources (through as many layers of virtualization as needed)
 - For example: Customer is notified that particular physical disk is receiving intermittent errors. Customer would like to understand which virtual machines would be effected if the disk failed.

Virtual System Modeling Basics

- **Host System (or Host Computer System)** – In a virtualized computer system environment the computer system that contains resources from which Virtual Systems are constructed.
- **Virtual System (or Virtual Computer System)** – Computer Systems composed of partitioned, shared or virtualized resources presented from a host system. Terms also used for this concept are Virtual Machine, Hosted Computer, Child Partition, Logical Partition, Domain, Guest.
- **HostedDependency** is used to associate Virtual System with its Host System
 - HostedDependency **may** be used to associate “virtual” device with “host” device.
- **LogicalIdentity** is used when simple direct host device allocation is done to Virtual System (e.g. partitioning)



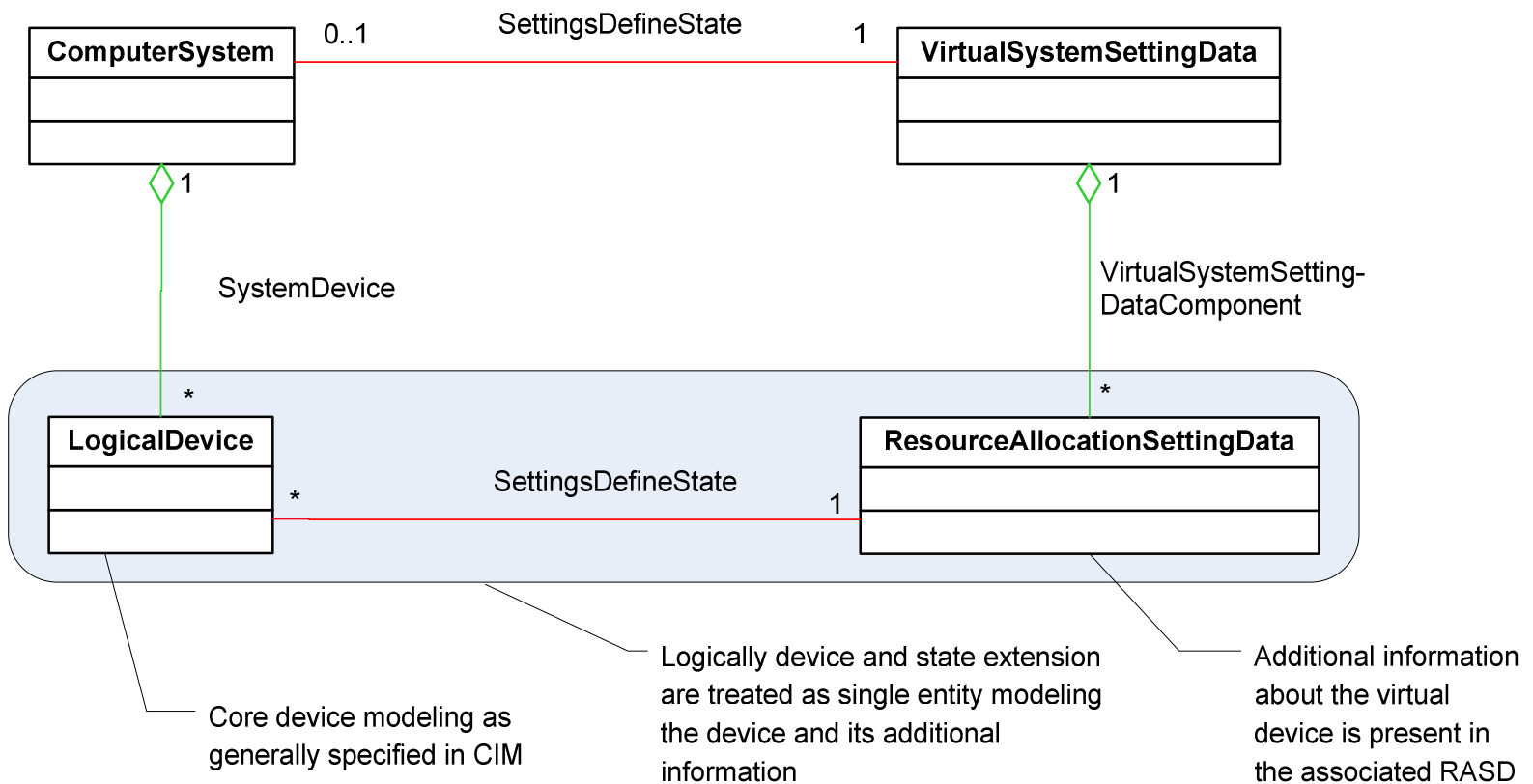
System Virtualization Related Profiles



Resource Virtualization

Virtual System Representation

Virtual System State Extension

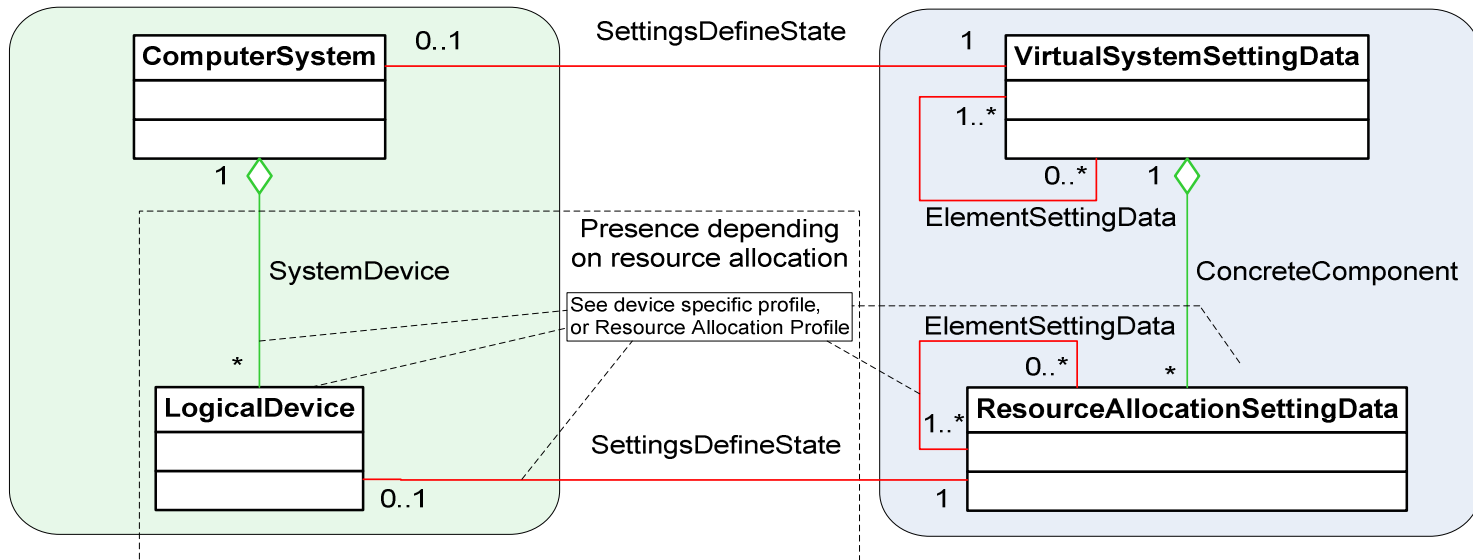


Resource Virtualization

- **Basic Principle of System Virtualization modeling**
 - Devices represented by core CIM classes, additional information in associated setting data
- **Virtual System Configuration defines virtualization extensions**
 - Also used as input for Virtual System creation
 - Recorded and Active State
 - Snapshots

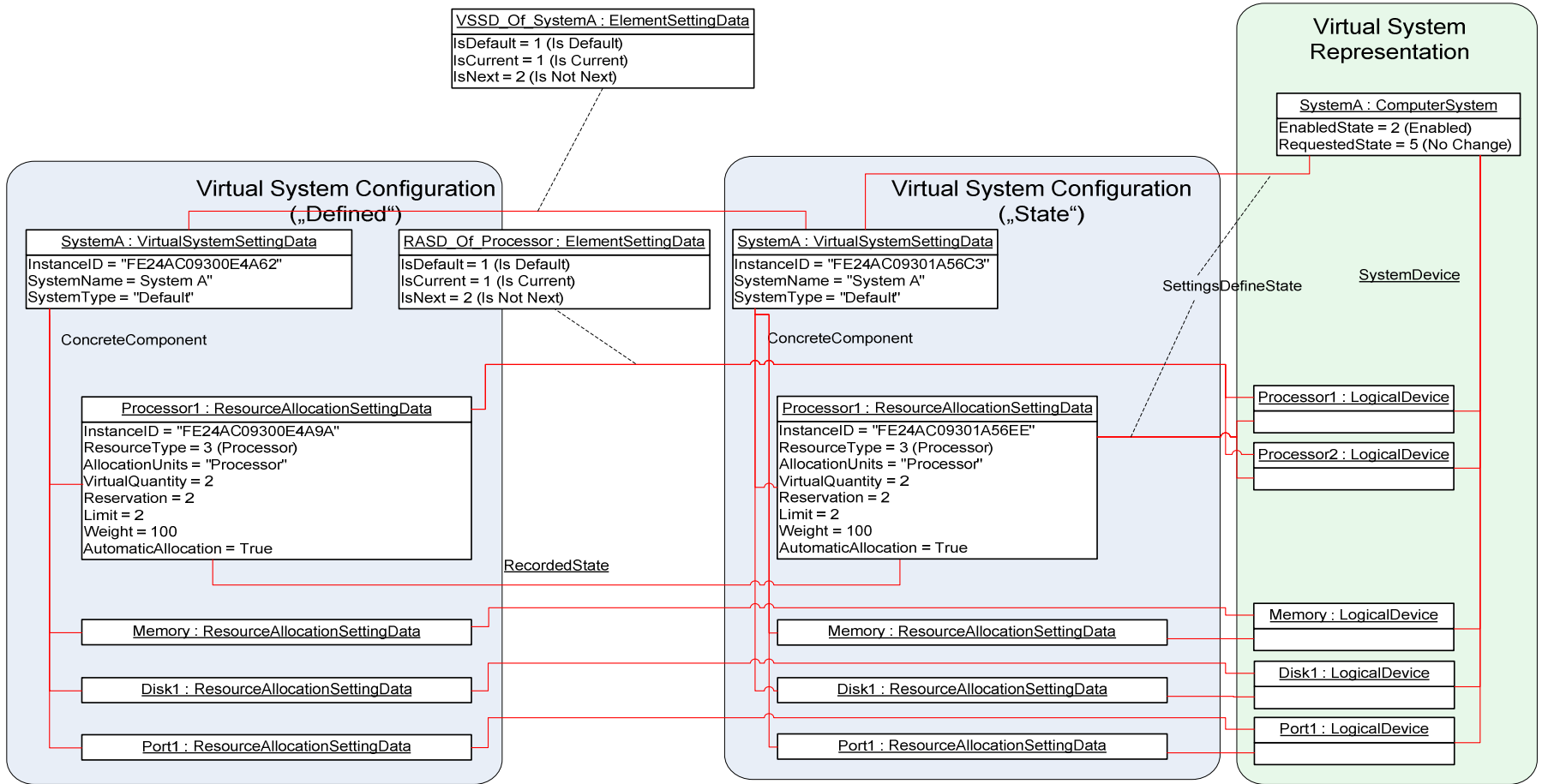
Virtual System Representation

Virtual System Configuration



- **RASD -- Key Class for describing aspects of virtualization**
 - Used on create request
 - Used to represent settings specifically related to virtualized resource Resource Type -- The type of resource this allocation setting represents
 - Used in SettingDefinesCapabilities association
- PoolID – ResourcePool allocated from or to be allocated from
- ConsumerVisibility {Unknown, Passed-Through, Virtualized}
- HostResource[] – exposes specific assignment to host or underlying resource
- AllocationUnits
- VirtualQuantity
- Reservation – Amount of resource guaranteed to be available for this allocation
- Limit – Upper bound of resource that will be granted for this allocation
- Weight – relative priority for this allocation
- AutomaticAllocation/AutomaticDeallocation – whether resource is automatically allocated at power on/deallocated at power off
- Parent – Parent of resource, for example controller for port
- Connection – the thing to which this resource is connected – for example named network
- Address – for example MAC address
- MappingBehavior – How this resource maps to underlying resources {Dedicated, Soft Affinity, Hard Affinity, Not supported}

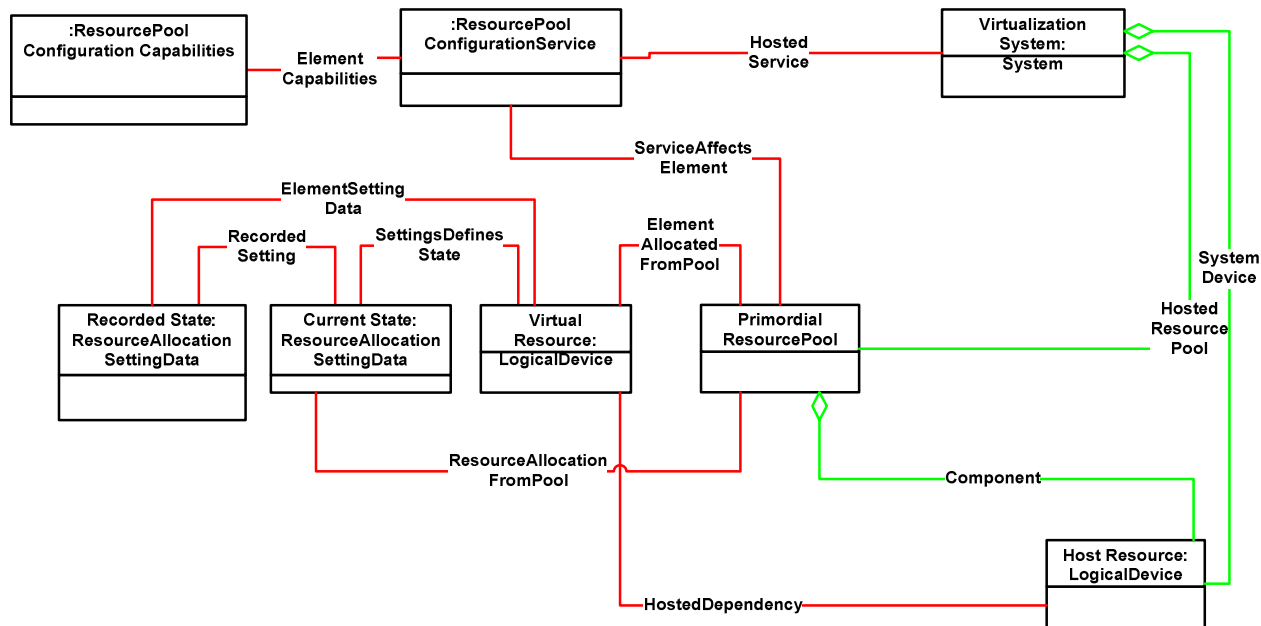
System Configuration



Resource Virtualization Pattern

ResourcePool abstracts the resource available to the virtualization layer for assignment or allocation to a hosted computer system

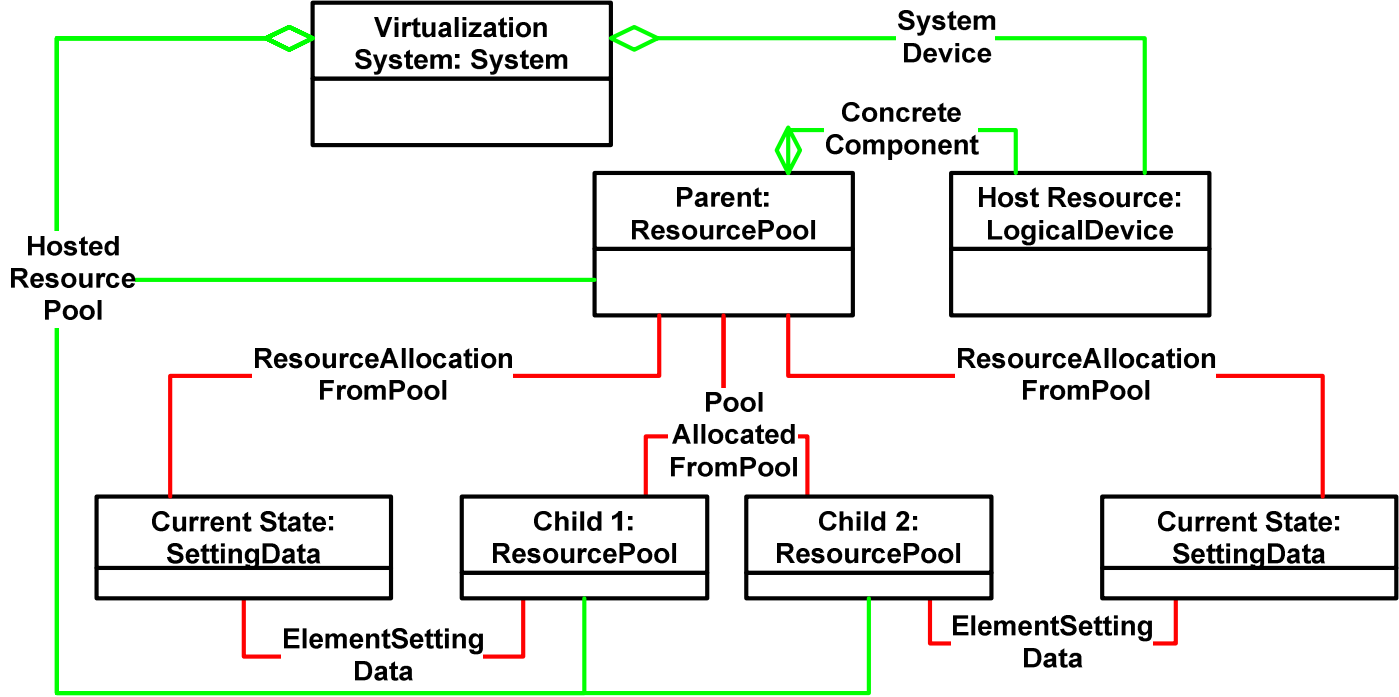
- Virtual Computer System resources are then created/assigned from resource pool on hosting system.
- Component Association to component resources
- ResourcePool may have no member resources (for example a pool that enables creation of virtual Ethernet adapters)
- **ResourceAllocationSettingData** represents the assignment of resources to hosted computer system. Specifies amount of resource allocated from Host System, and amount presented to Virtual System
 - One instance records persisted or recorded state (typically reflecting values in a config file)
 - One instance records current running values (when resource is allocated)
- **AllocatedFromPool** relationship between ResourcePool on hosting system and “VirtualResource” of virtual system



- Key Properties
- Primordial – Always exists and aggregates host resources
- Capacity
- Reserved
- ResourceType, subtype, units

Hierarchical Resource Pools

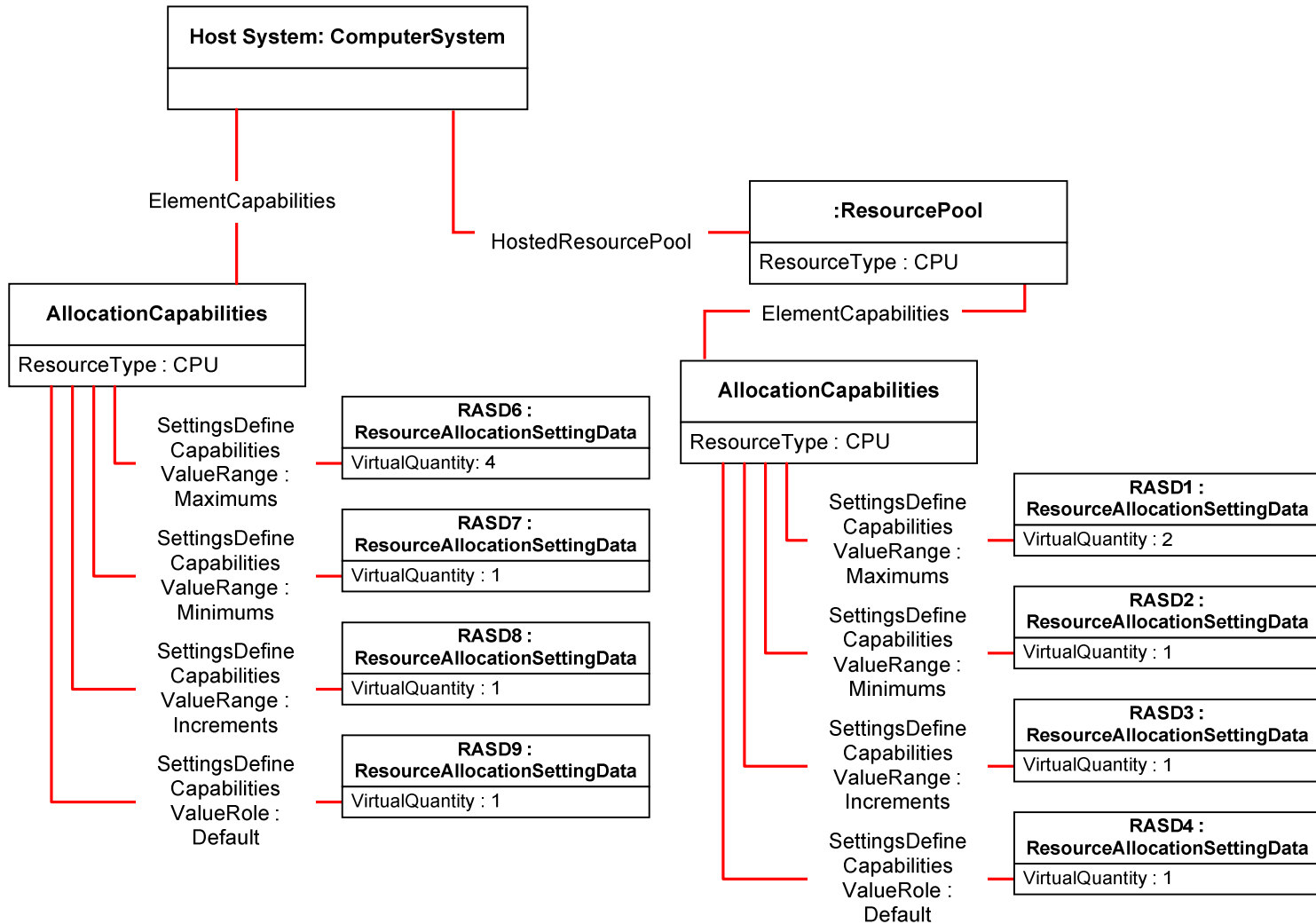
- Optional capability to enable parent and child pools for additional resource management
- Pool management service for pool hierarchy management.



Capabilities and Settings

- Use new “SettingsDefinesCapabilities” pattern for specifying details of various capabilities.
 - System Level capabilities, Resource Pool capabilities, Virtual Machine Capabilities
- For each resource pool an AllocationCapability instance and a collection of ResourceAllocationSettingData instances provide information about the capabilities of the virtualization layer for the particular resource.
- AllocationCapability includes information on how resources are allocated from the associated pool
- SettingsData instances describes values (min, max, default, increment) for allocation of resource from resource pool, and presentation of resource to virtual system
 - ResourceType, Reservation, Weight, Limit, ResourceUnits, VirtualQuantity, VirtualUnits

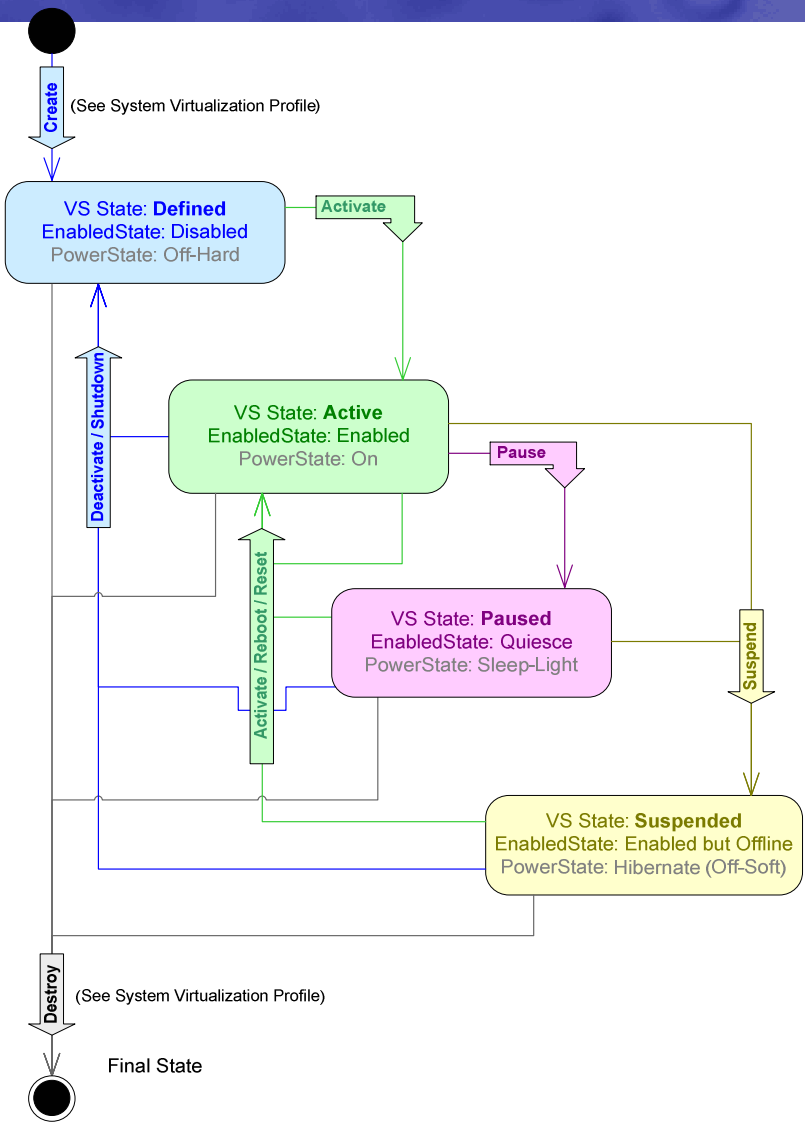
Capabilities and Settings



VirtualSystemManagement Service

- DefineSystem() -- Defines a virtual system. Input that is not completely specified will be filled out with default values
 - Embedded instance of class CIM_VirtualSystemSettingData that is used to define attributes of the virtual system to be defined
 - Array of embedded instance of RASD that describe desired resources
 - Reference to VSSD that refers to virtual system configuration used to complement the configuration of new virtual system if parameters in VSSD and RASD are not provided.
- DestroySystem() – Destroys Virtual System
 - Input is reference to CS instance
- AddResourceSettings () – Adds resource to virtual system configuration – if virtual system is active adds to virtual system
 - Array of embedded instance of class CIM_ResourceAllocationSettingData (RASD) that describes resources to be added to the virtual system
- ModifyResourceSettings()
 - Array of embedded RASD instance for each resource to be modified
- ModifySystemSettings() -- Modifies virtual system settings
 - Input is ref of VSSD to be modified and instance for modified values
- RemoveResourceSettings() -- Removes virtual resource settings from virtual system
 - Input is array of references to RASD representing resources to be removed
- All methods have capability of returning job if long running

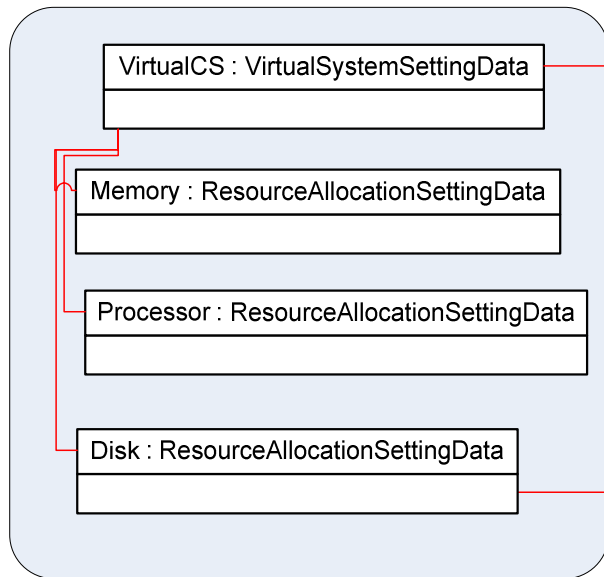
Virtual System State



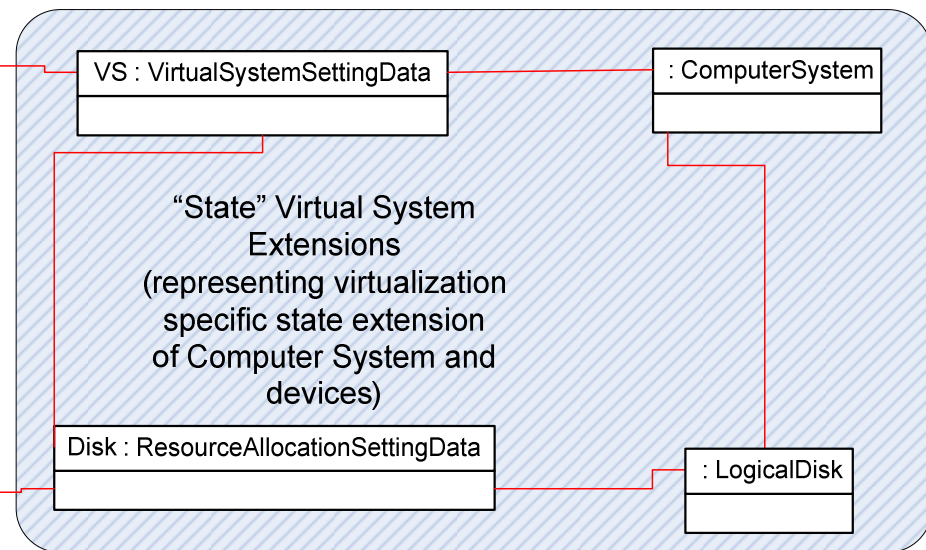
- **“Defined” State**
 - Virtual computer system is defined at the virtualization platform, but not yet instantiated.
 - There is an instance of class CIM_ComputerSystem in this state.
 - A virtual system in the “Defined” state is not enabled to perform tasks.
 - Typically in this state the virtual system does not consume any resources
- **“Active” State**
 - Virtual computer system is instantiated at the virtualization platform and its resources are performing tasks. .
- **“Paused” State – Optional**
 - Host resources remain allocated
 - Virtual system not enabled to perform tasks.
- **“Suspended” State – Optional**
 - Virtual resource persisted
 - Virtual resources represented by device instances but host resources may have been deallocated.

Virtual System State “Defined”

“Defined”
Virtual System Configuration
(representing Virtual System Definition)



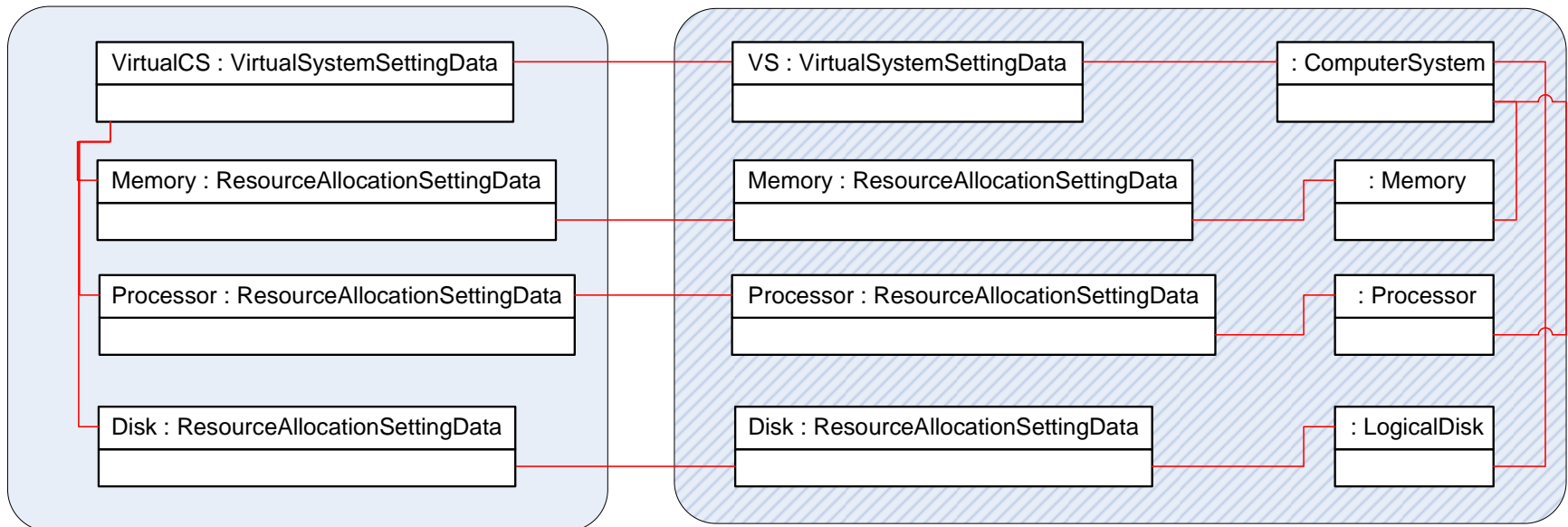
Virtual System Representation
(representing Virtual System Instance)



Virtual System State “Active”

“Defined”
Virtual System Configuration
(representing Virtual System Definition)

Virtual System Representation
(representing Virtual System Instance)



Virtualization Management Scenarios

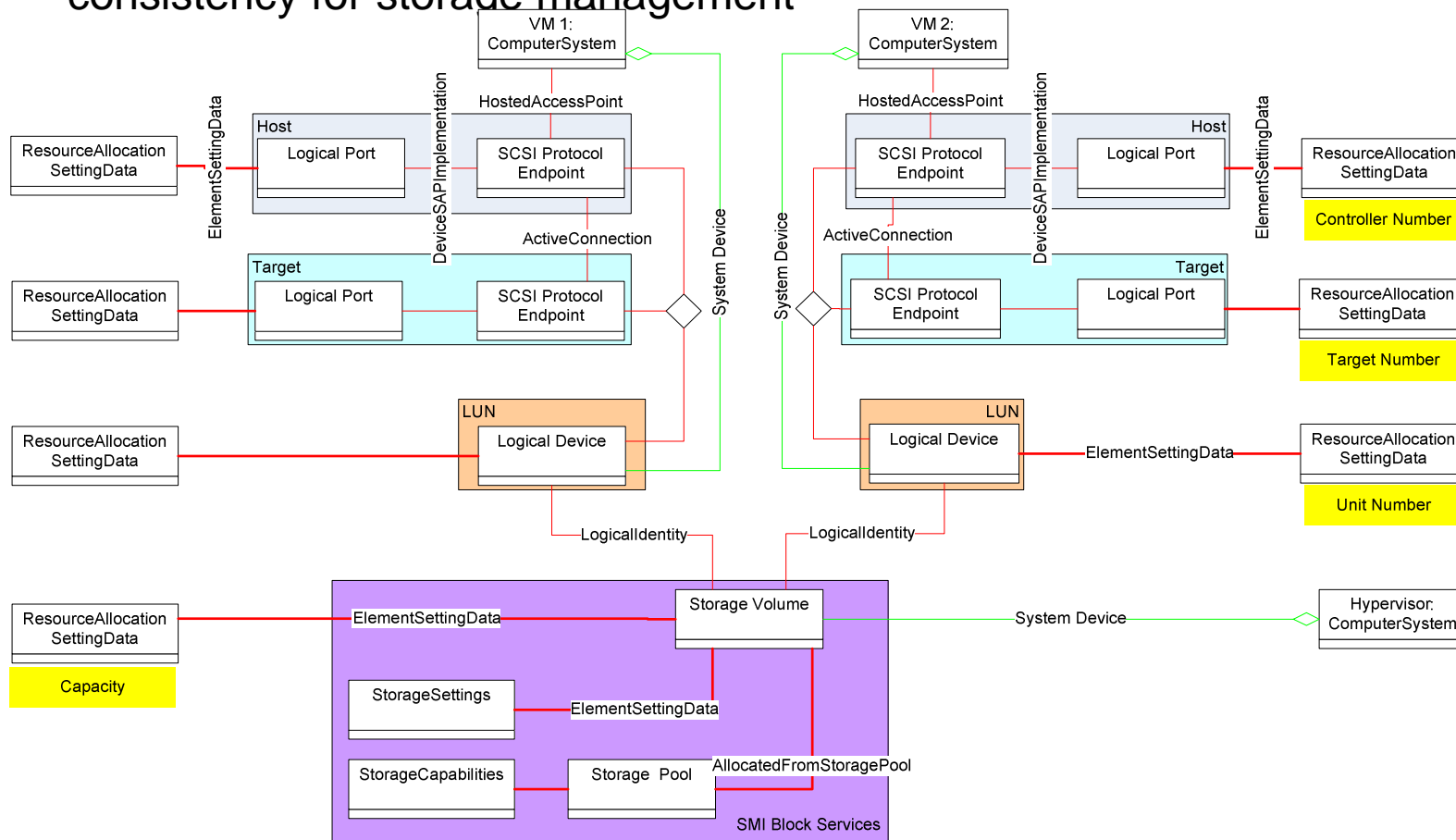
- **Managing the Host Computer System**
 - Discovering the CIM implementation for a System Virtualization
 - Discovering a Host Computer System
 - Determining the Capabilities of an Implementation
 - Determining the Supported Resource Types of an Implementation
 - Finding Resource Pools and their Constituent Resources
 - Determining the Capacity and Allocation of a Resource Pool
 - Determining Resources Allocated from a Resource Pool
 - Determining the Valid Settings for a Resource Allocation
 - Locating Virtual Systems Hosted by a Host Computer System
- **Managing a Virtual Computer System**
 - Creating a Virtual Computer System
 - Determining a Virtual System's State and Other Properties
 - Determining the "Defined" Virtual System Configuration
 - Determining the Virtual System Structure
 - Changing the Virtual System State
 - Modifying a Virtual System
 - Destroying a Virtual System
 - Managing Snapshots
 - Creating a Snapshot
 - Locating Snapshots of a Virtual System
 - Locating the Most Current Snapshot in a Branch of Snapshots
 - Locating Dependent Snapshots
 - Applying a Snapshot
 - Destroying a Snapshot

OnGoing Work

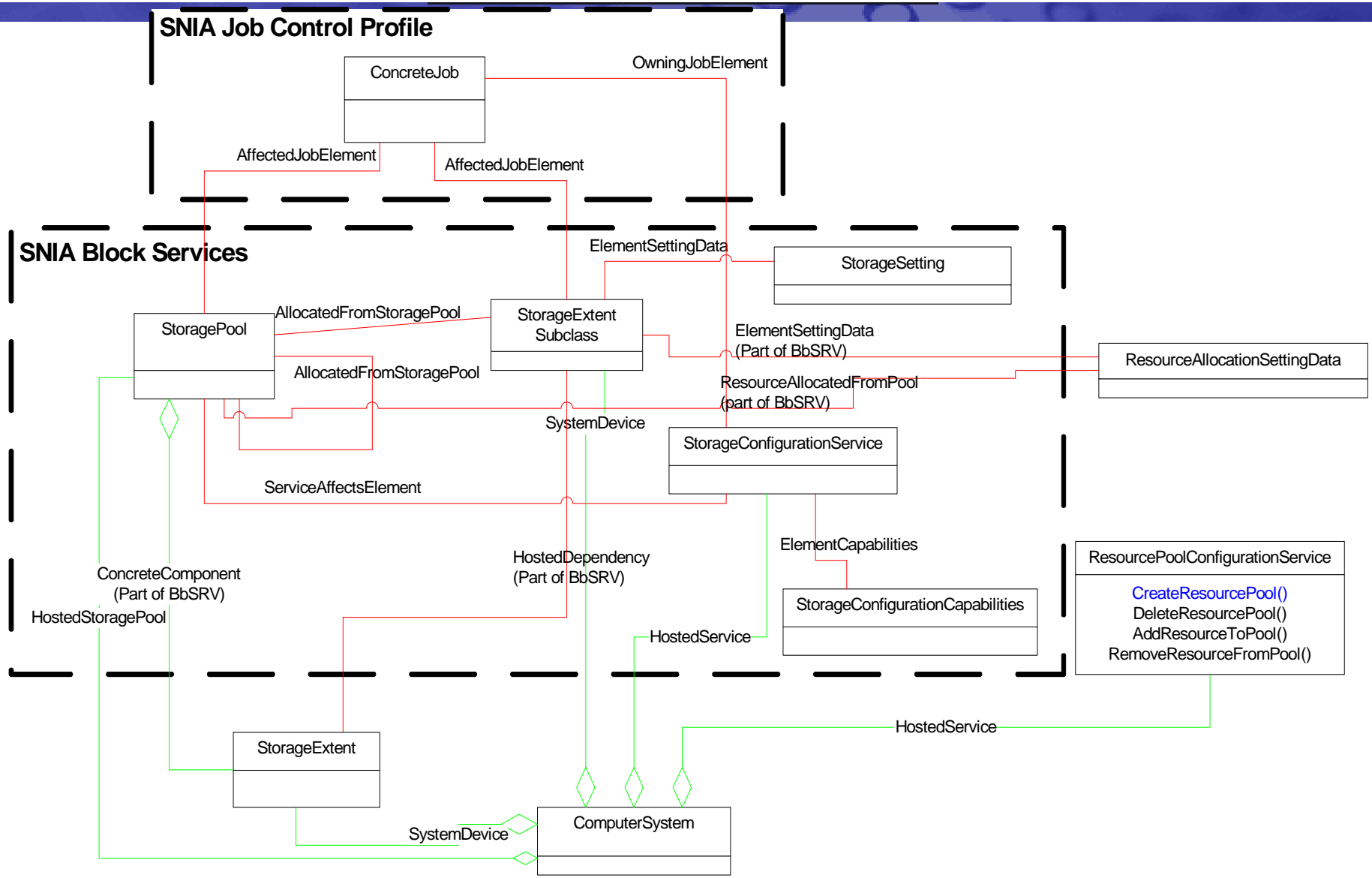
- Networking
- Storage
- Image Formats (OVF) and Image Management

Virtual Storage and Adapter Modeling

- Block and File based Virtual Disks
- Leverage SNIA profiles and packages to provide management application consistency for storage management



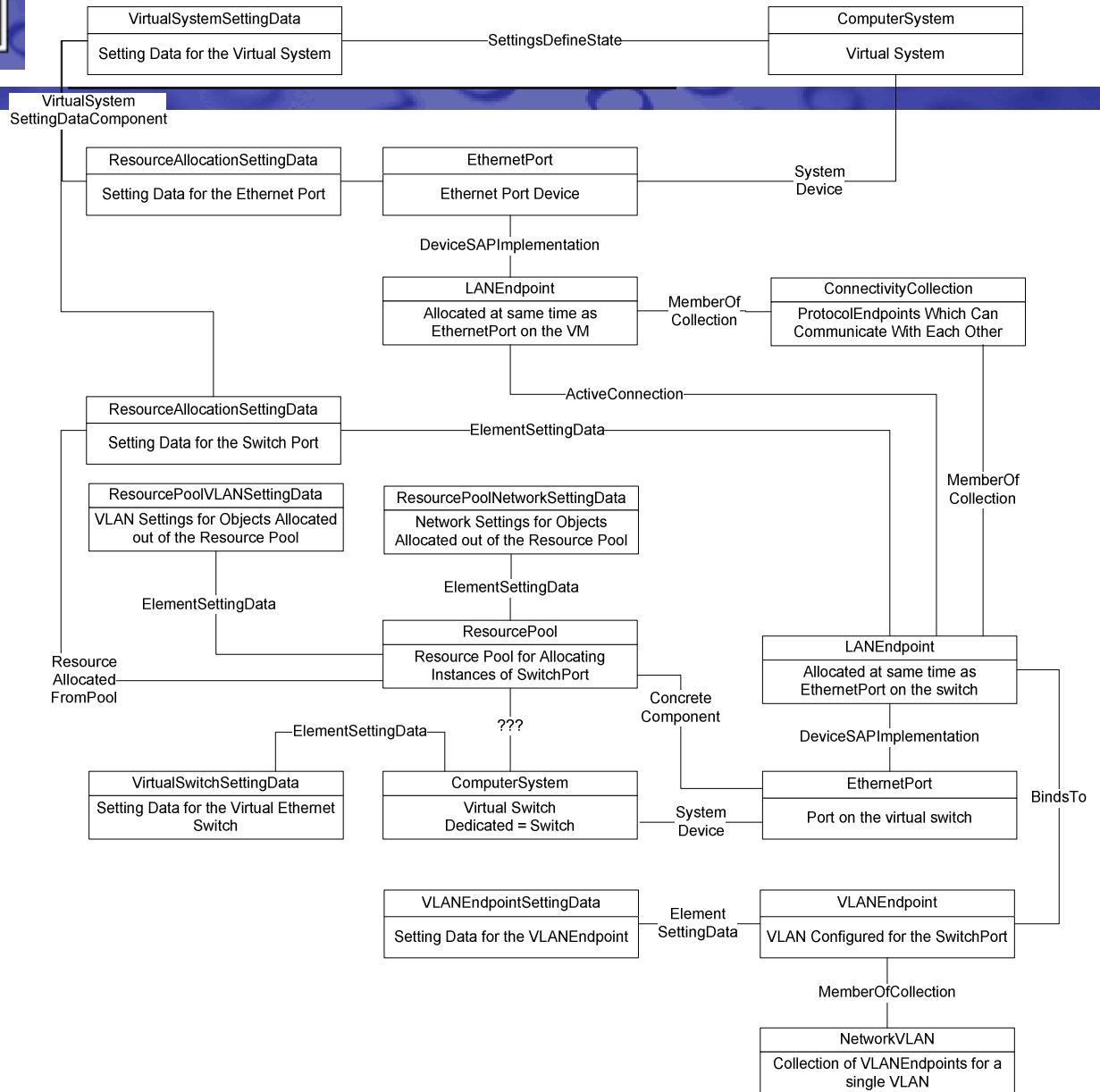
Block Based Storage Resource Virtualization – Instance Diagram



Networking

- Systems have internal virtualized network
- Objective is to leverage existing modeling work as much as possible.
- First phase is to model virtual ethernet adapter, virtual switch and enough of virtual network to display VM connectivity.

Networking



Future Work

- Profiles for Additional Virtual Devices
 - Keyboard/Mouse/Video
 - CD-ROM, Floppy – want to model both the (virtual) device, and ability to map to (virtual) media (ISO image, file, real media...)
 - Sound, Video, Serial, USB

Summary

- Flexible Model for Virtualization & Partitioning
 - Schema Changes part of CIM 2.16 & 2.17
- Profiles available for public comment and feedback per DMTF process
- Whitepaper published
- Plugfest at ManDevCon
- Virtualization DMTF announcement at Gartner IT Conference
- Network and Storage as next focus areas