

November 15-18, 2010



Santa Clara Marriott
Santa Clara, CA

Profile Usage Guide 1.1: An update

Michael Johanssen
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.



Overview PUG 1.1

- Management domain and managed object types
- Managed environment and managed objects
- Profiles
- Adaptations
- Requirement levels
- Conditions
- Features
- Error reporting requirements (standard messages)
- Indications
- Metrics
- Merge algorithm

Profiles

- A profile
 - defines a *management interface* between implementations of a WBEM server and a WBEM client and/or WBEM listener
 - defines a *model* and its behavior in context of a *management domain*
 - management domain
 - establishes for what the model is defined
 - expressed in terms of managed object types
 - usually term definitions, detailed in the Description clause
 - model
 - a set of adaptations that establish the semantics of the management interface by defining
 - » how managed objects and their relationships are to be represented in CIM
 - » how managed objects behave if modifications are initiated through the model



Management domain

Managed environment

- **Management domain**
 - area of work or field of activity with common management requirements, common terminology, and related management functionality
 - an abstraction, composed of types of managed objects
 - Examples: File management (FM), system virtualization (SV), ...
 - can be subdivided:
 - FM: File representation, --access, --locking, ...
 - SV: Virtual system representation, --state management, --resource management, ...
- **Managed environment**
 - A concrete occurrence of a management domain
 - Profiles are implemented for one or more types of managed environments
 - Examples:
 - FM: Linux file management, Windows file management, z/OS file management
 - SV: VMware based system virtualization, HyperV based ~, Xen based ~, KVM based ~, z/VM based ~
 - The set of managed objects exposed and controlled by an implementation
 - Examples:
 - FM: The filesystems on "server1.acme.com"
 - SV: The virtualization platforms on host "host1.acme.com"
- **Managed object**
 - a physical entity, a service, or other kind of resource that is managed
 - Examples:
 - FM: File "/home/user1/.profile" on "server1.acme.com", lock on "c:\share\fs1\file2", ...
 - SV: Host "host1.acme.com", virtual system "VS1" on "host1", virtual disk "C" of "VS1" on "host", ...

Profile elements

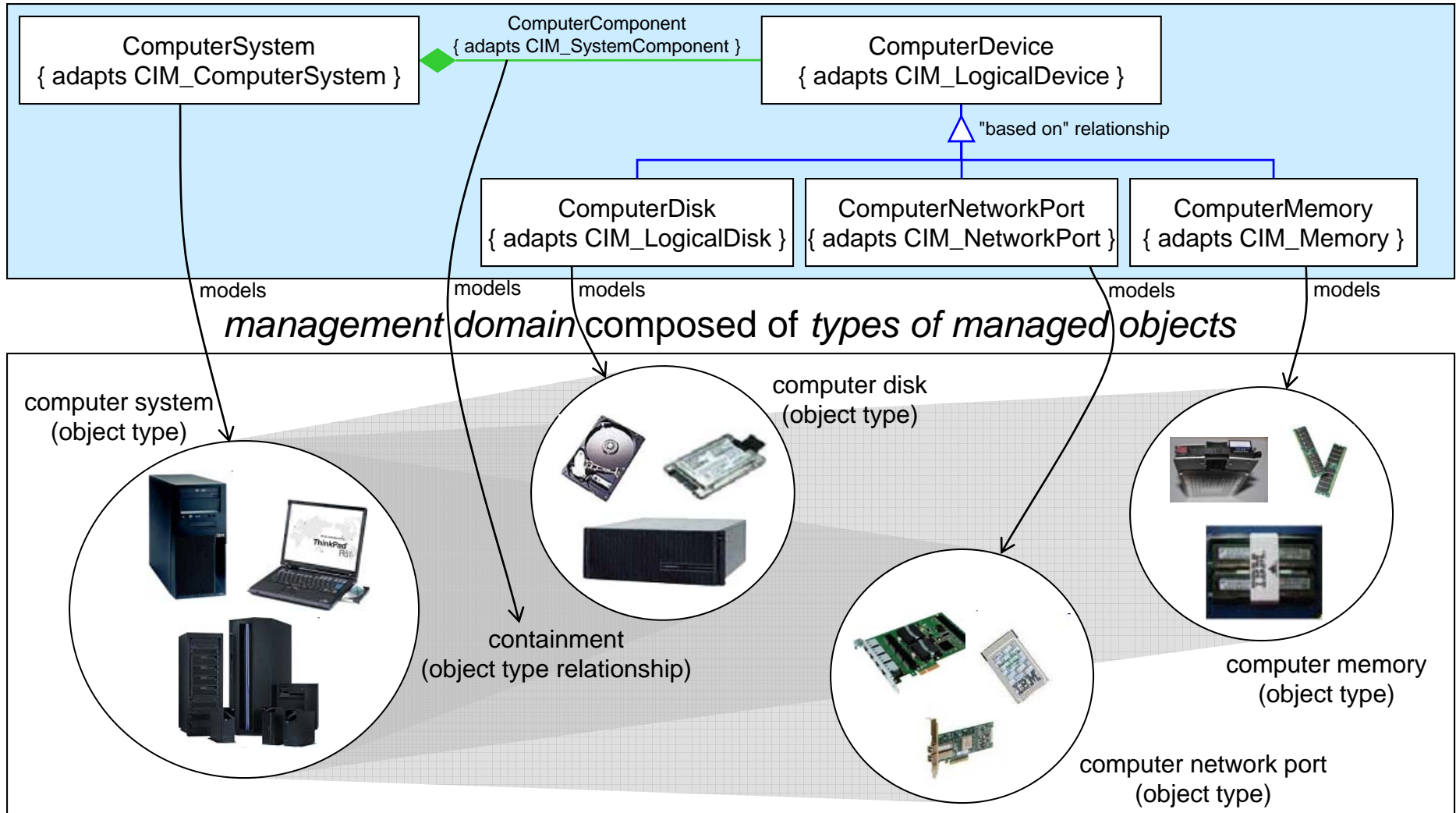
- **adaptations** (replacement for "profile classes")
 - management domain context
 - property requirements
 - method requirements
 - operation requirements
 - error reporting requirements
 - instance requirements (adaptations of ordinary and association classes)
 - indication generation requirements (adaptations of indication classes)
- **features** (enables grouping of requirements)
- **profile references** (extended to allow for multiple references to the same profile)
- **registry references** (new concept to provide for the integration of xml based registries)
 - message registries (messages used for error reporting, and for defining indications)
 - metric registries
- **events** (if indications are defined)
- **use cases** (extended, now formally stating preconditions and postconditions)
 - describe the client use of the profile defined model

Adaptations

- Adaptation
 - models (an aspect of) a managed object type
 - defines a *use of a schema defined class* for a particular *purpose*
 - implies the requirements of the schema definition of the adapted class (and its base classes)
 - can be *based on* other adaptations (with certain restrictions)
 - implies the requirements of the base adaptation(s)
 - defines (additional) requirements and constraints on top of the schema definition
 - *property* requirements
 - *method* requirements
 - method semantics – effects in the managed environment
 - error situations and CIM status codes
 - standard messages (optional)
 - *operation* requirements – imply the requirements of the operations specification
 - operation semantics – effects in the managed environment
 - error situations and CIM status codes
 - standard messages (optional)
 - defines *instance requirements* that define how and when managed objects and their relationships are to be represented by adaptation instances

Modeling with adaptations

CIM model composed of adaptations



Adaptations

Delta requirements

- Basis for an adaptation
 - Schema definition of the adapted class
 - Operation definitions in the selected operations specification
 - Metric definitions
 - Message definitions
- Adaptations define "delta" requirements
 - beyond those defined in the schema
 - beyond those defined in base adaptations
 - beyond those defined in referenced profiles
 - beyond those defined in registry elements
 - beyond those defined in registry schemas
 - beyond those defined in operations specifications
 - ...

Adaptations

How do refinements work ?

- Refinements can only strengthen requirements
- Adaptations
 - requirement level
 - management domain context
 - instance requirements
- Property requirements
 - requirement level
 - value constraints
 - management domain context
- Methods & operation requirements
 - requirement level
 - parameter value constraints
 - management domain context / semantics
- Error reporting requirements
 - standard messages
 - CIM status code

Requirement levels

- **Mandatory**
 - functionality required for a functioning implementation of the profile
- **Optional**
 - functionality not required, but is auxiliary or complementary for a functioning implementation of the profile
- **Conditional**
 - functionality is required for a functioning implementation only under certain conditions, but is otherwise optional
- **Conditional exclusive**
 - as conditional, but in addition, the functionality shall not exist if the condition is not true
- **Abstract** (applicable to adaptations only)
 - functionality is not independently required for a functioning implementation, but instead is designed to be required in the context of other class adaptations that define the abstract class adaptation as a base adaptation

Adaptation example: VirtualSystem adaptation I

6 Description

...

Definition of a managed object type

- Usually done by means of term definitions and prose text in the Description clause.
- Should enable an implementer to recognize respective types of things in "his" implementation environment
- Frequently refers to other terms used in the industry for the same thing

6.2 Virtual system

A virtual system is a virtualized (computer) system that is composed of partitioned and/or virtualized computing resources and/or system devices. Other common industry terms for such a system include: Virtual machine, hosted computer, child partition, logical partition, domain, guest, and container. In some virtualization environments, a virtual system may act as the host system to other nested virtual systems, such as a logically partitioned system with each partition running a separate virtualization platform.

6.3 Virtual resource

....

Adaptation example: VirtualSystem adaptation II

5 Synopsis

Local profile reference name; needed because a particular profile can be referenced more than once for different purposes.

Table 2 — Related profiles

Profile name	Profile name	Organization	Version	Relationship	Description
...
ProfileRegistration	Profile Registration	DMTF	1.1	Mandatory	Registration of profiles. For details, see DSP1033.
...

adapts the CIM_ComputerSystem class; this implies requirements from the CIM schema definition

Table 2 — Adaptations

Adaptations	Adaptations	Requirements	Description
...
VirtualSystem	CIM_ComputerSystem	Mandatory	See 7.2.3
VirtualSystemState	CIM_VirtualSystemSettingData	Mandatory	See 7.2.4
StateOfVirtualSystem	CIM_SettingsDefineState	Mandatory	See 7.2.5

Adaptation Example: VirtualSystem adaptation III

7 Implementation

...

7.2.3 VirtualSystem: CIM_ComputerSystem

7.2.3.1 General

The VirtualSystem adaptation models virtual systems; virtual systems are described in the Profile Registration profile (see DSP1033). The VirtualSystem adaptation adapts the CIM_ComputerSystem class and is based on the CentralElement adaptation defined in the Profile Registration profile (see DSP1033).

7.2.3.2 Instance requirements

If represented, virtual systems shall be represented by VirtualSystem instances.

7.2.3.3 Element requirements

7.2.3.3.1 General

Table 22 — VirtualSystem: Element requirements

Elements	Requirements	Description
Base adaptations		
ProfileRegistration:CentralElement	Mandatory	See DSP1033.
Properties		
EnabledState	Mandatory	
Methods		
RequestStateChange()	Conditional	
Operations		
GetInstance()	Mandatory	See DSP0200.
EnumerateInstances()	Mandatory	See DSP0200.
EnumerateInstanceNames()	Mandatory	See DSP0200.
ModifyInstance()	Optional	See 7.2.3.2.4, and DSP0200.

Description states which managed object type is modeled

Carefully phrased, avoiding the requirement that *all* virtual systems existing in the ME must be represented.

Adaptation instances conform to all requirements of the adaptation, its base adaptation(s) and the adapted class

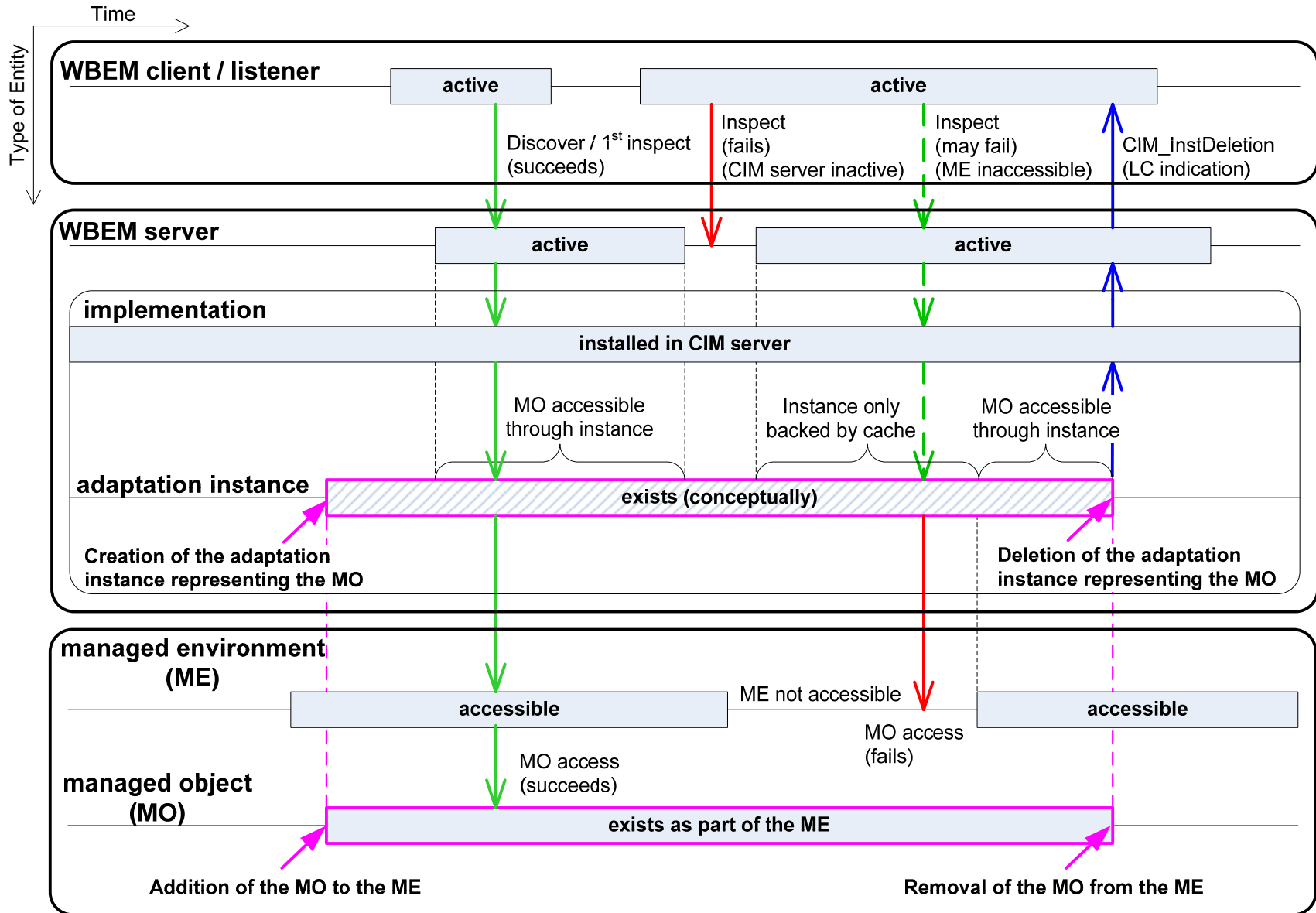
All requirements (base adaptations, properties, methods and operations) listed in one table

based on the CentralElement adaptation defined in the Profile Registration profile; base adaptations imply additional implementation requirements, such as - in this example - the relationships required by profile advertisement methodologies

Adaptation instances

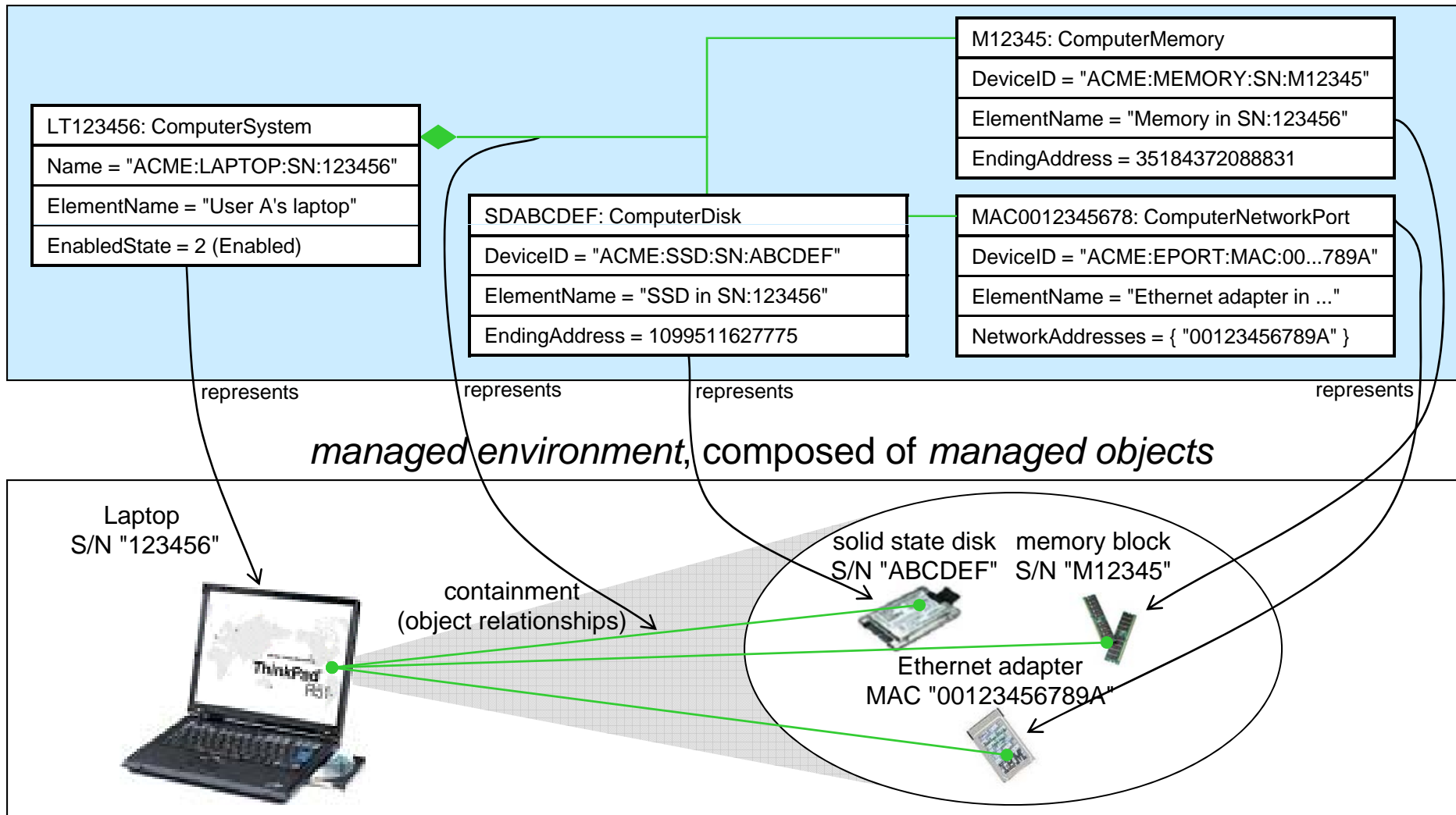
- Adaptation instance
 - instance of the adapted class that conforms to the requirements of the adaptation
 - adaptation instances always represent (aspects of) managed objects
 - adaptation instances are not "free" like database records
 - they are always bound to the state of the represented managed object
 - however, the represented managed object state is not required to be accessible all time – it may be cached
 - to which extent the state is current depends on the implementation
 - the state of the managed object can be manipulated by executing methods and operations
 - an adaptation instance is always required to represent the MO state – in both cases: Success or failure
- An adaptation instance exists as long as the represented managed object exists within the managed environment.
 - Existence of instances is a concept only, it does ***not imply*** an instance store or repository

Existence of adaptation instances



Representing with adaptation instances

Adaptation instances: CIM instances that conform to adaptations



Error reporting requirements

Cascade of messages, starting with a message required by the operations specification, and amended with messages defined in the message registry related to the profile.

"Mandatory" means that the set of messages is required if the error situation described by their combination occurs.

The description may give implementation directives about when to report the messages in the context of the management domain addressed by the profile.

Table 25 – VirtualSystem::RequestStateChange reporting requirements

Elements	Requirements	Description
WBEMMessageRegistry::WIPG0227, SVPCMessageRegistry::SVPC0500	Mandatory	Virtual system already in requested state.
WBEMMessageRegistry::WIPG0227, SVPCMessageRegistry::SVPC0501	Mandatory	Invalid state transition.

Indications

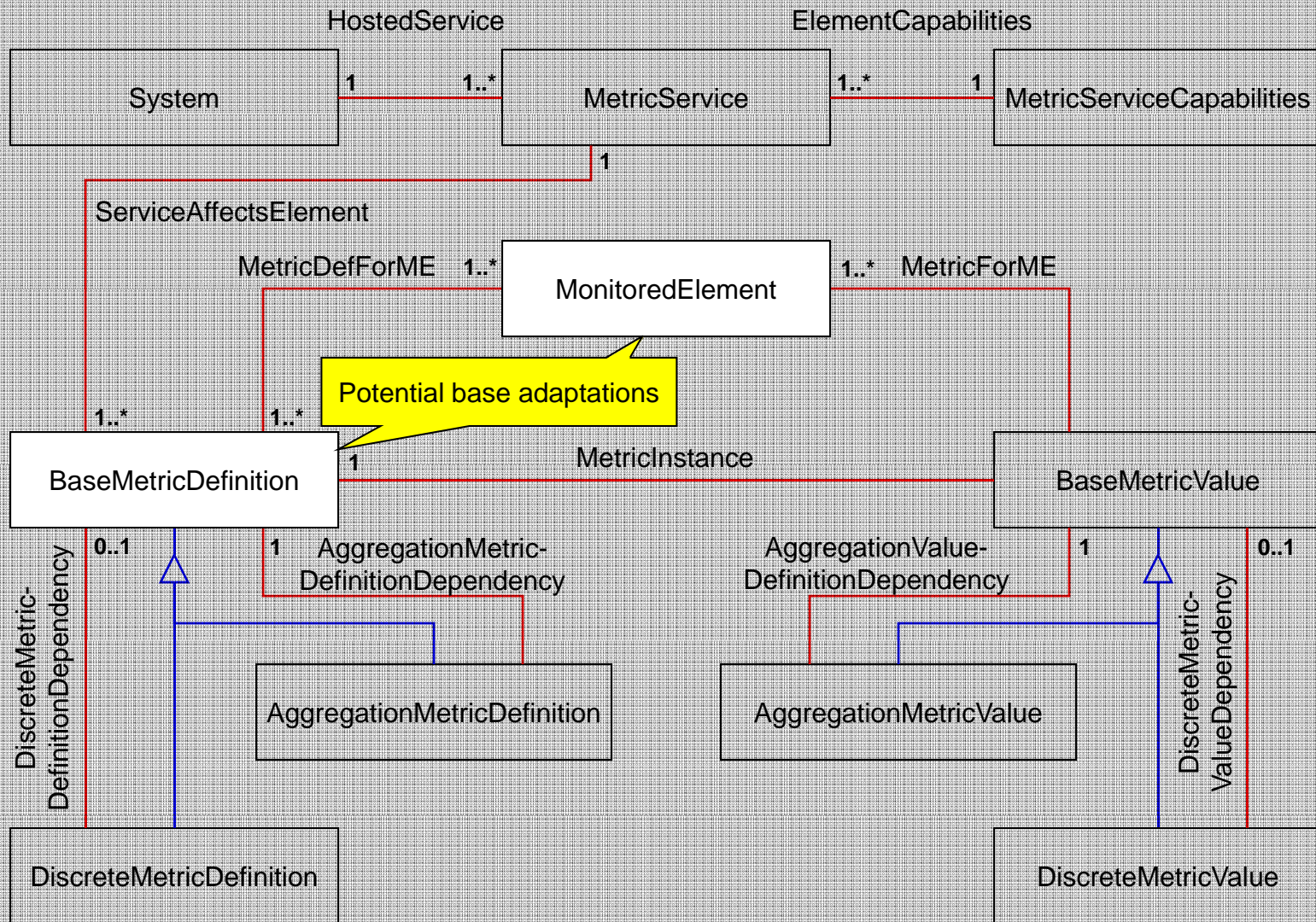
- Profiles defining indications reference the Indications profile (see DSP1054)
 - Referencing profiles do not need to define indication related infrastructure in profiles, such as indication filters, filter collections or subscriptions
 - This is addressed by the Indications profile
 - The Indications profile is implemented once per WBEM server
- Profiles base their indication adaptation on adaptations defined in the Indications profile
 - AlertIndication
 - The base for all alert indications defined in referencing profiles
 - Applies messages defined in a message registry
 - LifecycleIndication
 - The base for all lifecycle indications defined in referencing profiles

NOTE: Details about indications are provided in a separate presentation.

Metrics

- Profiles defining metrics reference the Base Metric profile (see DSP1053)
 - Referencing profiles do not need to define metric related infrastructure in profiles, such as metric definitions
 - This is addressed by the Base Metric profile
 - The Base Metric profile is implemented once per WBEM server
- Profiles base their metric adaptations on adaptations defined in the Base Metric profile
 - MonitoredElement
 - models the element in context of that metrics tbd.
 - BaseMetricDefinition
 - models the definition of a metric

Base Metrics model



Metric definition in profile

All metric concept related requirements covered by DSP1053

7.3.10 VirtualSystem

Table 10 - VirtualSystem: Element requirements

Element	Requirement	Description
Base adaptations		
ProfileRegistration:Central	Mandatory	See DSP1033
Base Metrics:MonitoredElement	Conditional	See DSP1053
Properties		
ElementName	Mandatory	See 7.3.10.1
...
Metrics		
SVPCMetricRegistry:CPUReserved	Conditional	See 7.3.10.5
SVPCMetricRegistry:CPUTime	Optional	See DSP8048 and DSP1053

Metric content covered by metric definition (in DSP8048), and metric xml schema

Individual requirement level per metric

Refinements in metric specific subclause

Merge algorithm I

- Profiles are typically not implemented "stand alone", but as part of the implementation of a set of profiles
- Profiles define "delta" requirements
- Additional sources for requirements are to be considered, such as the schema, the selected operations specification or message registries
- An implementation is required to conform to all of these requirements
- The PUG defines a "merge algorithm" that provides for collecting the base and delta requirements from the various sources
- Note that such a merge was already needed for profiles based on PUG 1.0; with PUG 1.1 a comprehensive approach is provided to solve that problem
- For the future it is envisioned that the merge process can be automated, using machine readable sources

Merge algorithm II

- The set of profiles to be implemented is determined by selection, and by profiles dependencies
- For each profile, implementers need to
 - determine conditions for conditional elements
 - select the optional elements to be implemented
 - this in turn might impact the set of profiles to be implemented
- Each profile defines requirements for one or more adaptations
- Each adaptation may be based on other adaptations defined in the same or in referenced profiles
- Dependencies may exist between adaptations
- As a result of the merge algorithm a set of "implementation-required" adaptations result, carrying the collected requirements from all sources
- Finally, implementers may choose to "combine" implementation-required adaptations if that is possible in a consistent way
- The final set of implementation-required adaptations could be regarded as an "implementation profile"

Merge algorithm III

A: Selected to be implemented

B: Referenced by A

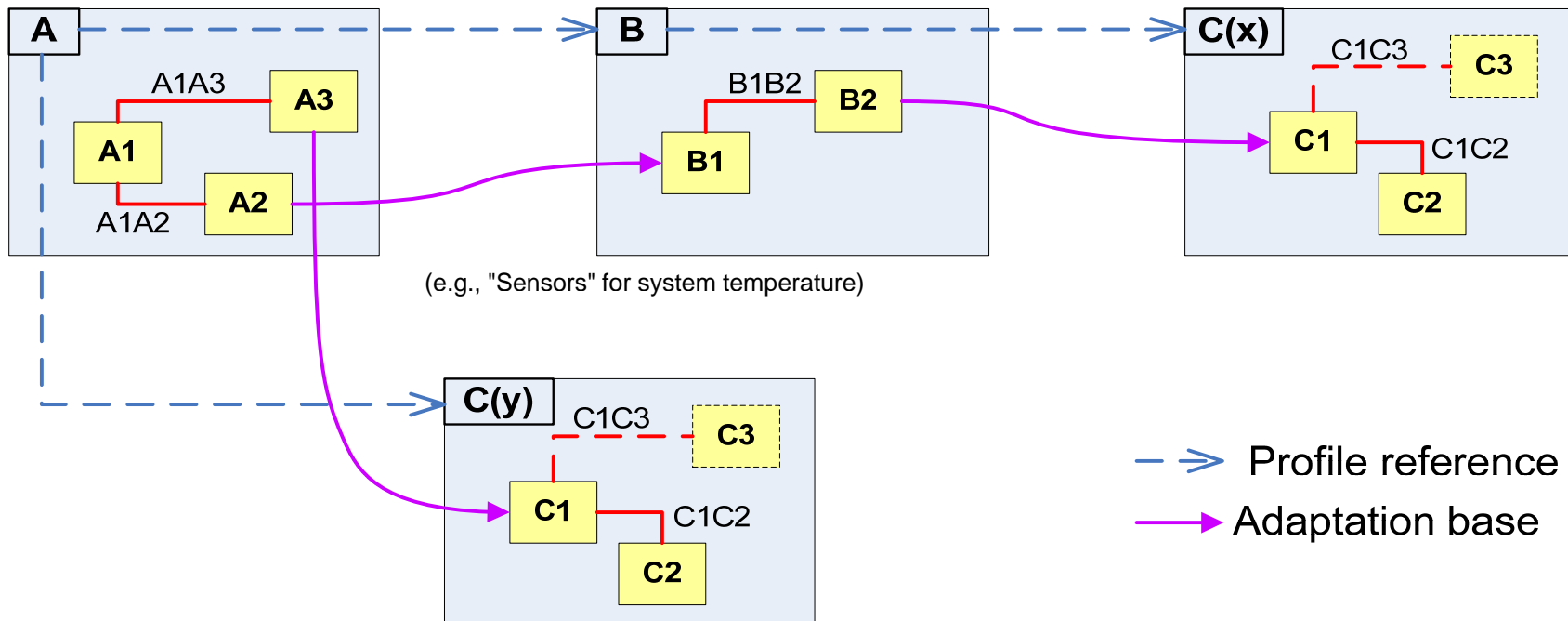
C: Referenced by A and C

Profile implementation set

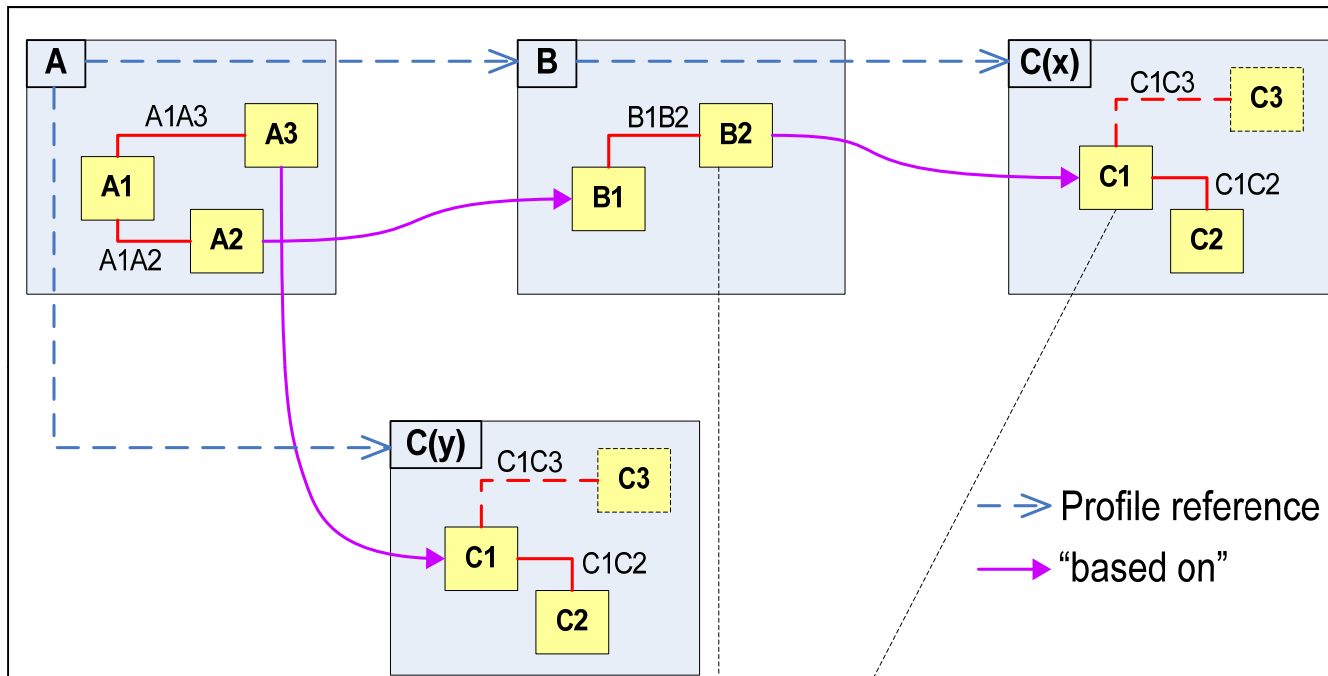
(e.g., "Computer System")

(e.g., "Fan")

(e.g., "Sensors" for fan speed)



Merge algorithm IV



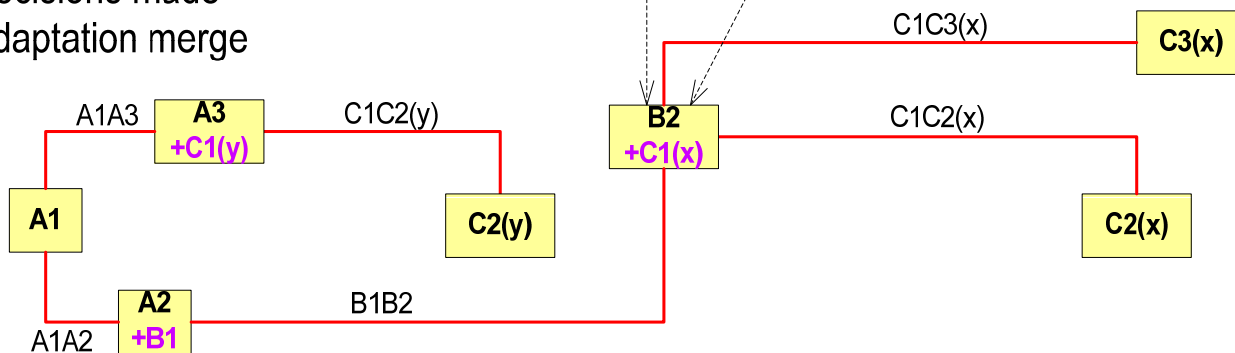
- - - -> Profile reference
 - - - -> "based on"

Resolve dependencies
 The result is a set of CIM classes that are to be implemented in accordance with the requirements imposed by the adaptations defined in the set of profiles.

Insofar, a set of CIM classes is implemented, not a set of profiles – but the CIM class implementations are required to conform with profile defined requirements.

Some adaptations are not implemented for themselves, but as part of adaptations that are based on them

Decisions made
 Adaptation merge



Status

- 1st Work-in-Progress published 2010-06-11
- 2nd Work-in-Progress imminent
- DMTF Standard imminent