WBEM/CIM Management

Basic concepts and availability in Fedora

Red Hat
Vítězslav Crhonek <vcrhonek@redhat.com>
June 13, 2011

Part I
Basic concepts

fedora

What is WBEM?

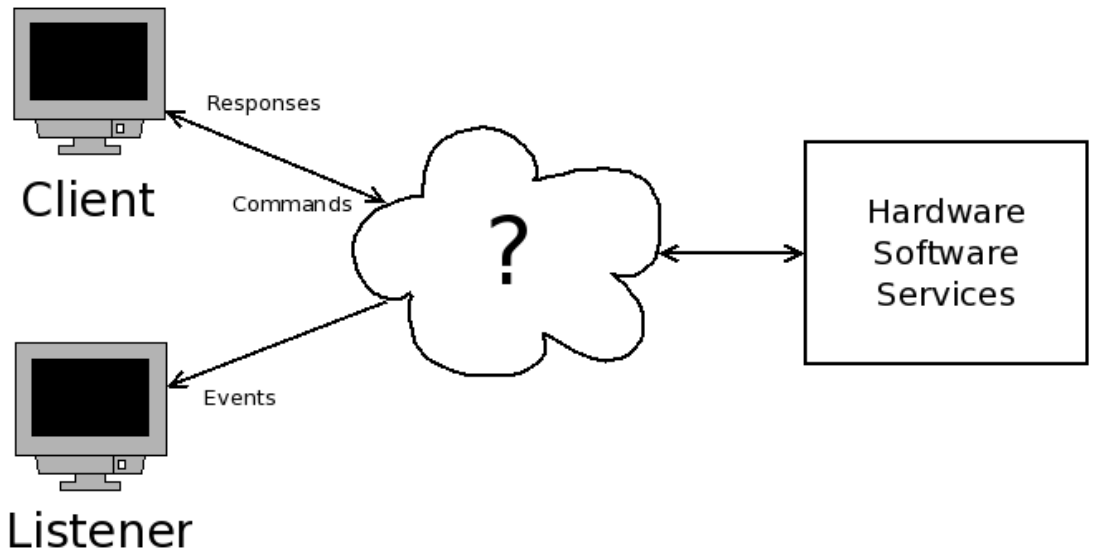
Section 1

Introduction

fedora

Management

- Common and flexible way of managing a collection of heterogenous devices and services
- Storage network, electrical power supply, desktop computing, telecomunication industries, etc.
- System administration - configuration, backup, user administration, security policies, performance monitoring, problem determination, etc.



fedora

DMTF

- DMTF - Distributed Management Task Force
- Industry body formed to lead the development, adoption, and interoperability of management standards
- ~160 member companies and organizations
- DMTF board of directors is led by 15 companies (AMD, Broadcom, CA, Cisco, Citrix Systems, EMC, Fujitsu, HP, Huawei, IBM, Intel, Microsoft, Oracle; Red Hat and VMware)
- `http://www.dmtf.org/home`

Section 2

WBEM architecture

fedora

WBEM

- WBEM (pronounced "web-em") - Web-Based Enterprise Management is End-to-End interoperable management suite designed by DMTF
- Emerged in mid (1996) to late 1990s, primarily for managing desktop systems, enterprise networks and E-business infrastructure, in late 1990s it started to evolve into more general-purpose management tool
- Major goals:
  - Reduced Total Cost of Ownership - interoperable management lowers the man-hours needed
  - Improved Time to Market - using standards
  - Reduced Development Time - existing information models can be used or expanded upon
  - Support for other management solutions - migration support for SNMP, DMI, etc.

fedora

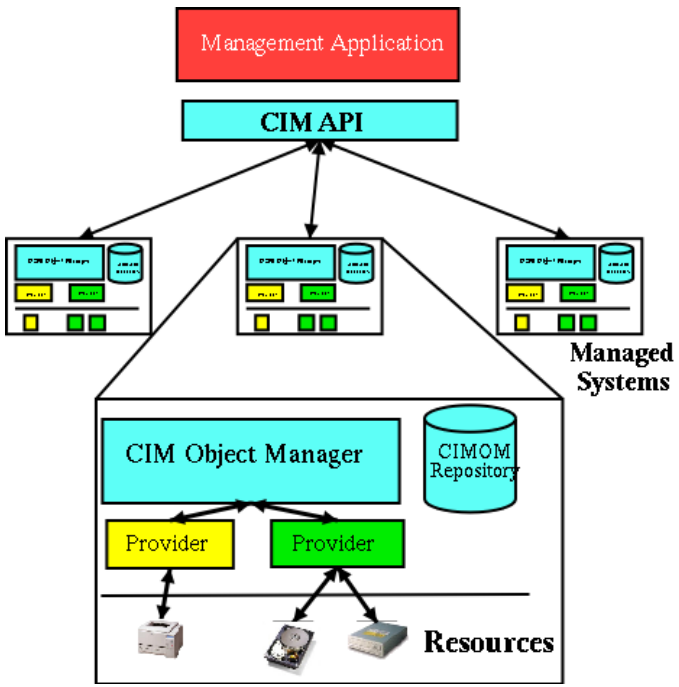
WBEM components

- WBEM consists of:
  - CIM specification (data modelling process and language)
  - CIM Server (broker between operators and managed systems)
  - CIM/XML encoding specification (representation of CIM in XML, encoding commands and responses)
  - CIM over HTTP access (transporting mechanisms for carrying commands and responses across a network)
- Key difference between WBEM and traditional management standards (e.g. SNMP, TMN):
  - Available modelling constructs (object-oriented language)
  - Clear separation of the interface used to acces information from the model of the device being managed
- `http://www.dmtf.org/standards/wbem`

fedora

WBEM Stack

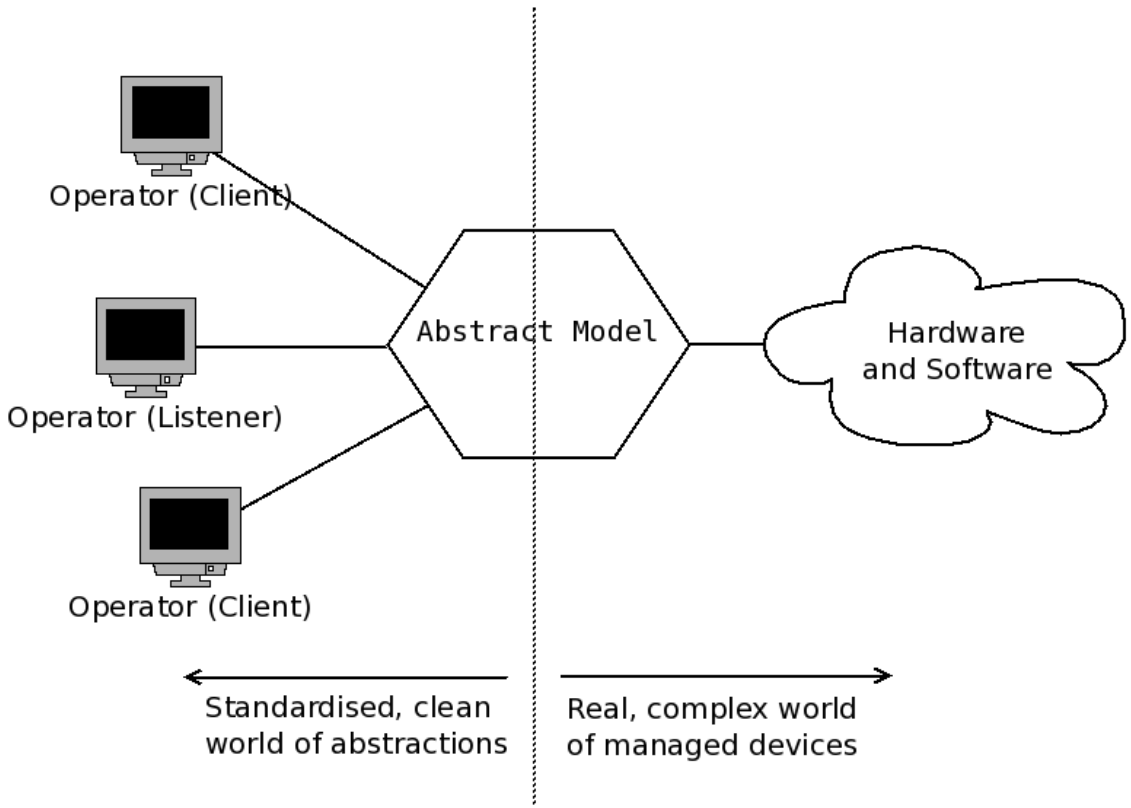
- CIM Client - a client that can format an XML document and send an HTTP request
- CIM Server (CIMOM) - an HTTP server that can decode XML and interface with providers
- Provider - a library that "knows" about the device/service being managed and can insteract with CIMOM



fedora

CIM

- CIM (Common Information Model) is language and methodology for describing management data standardized by DMTF, composed of a CIM Specification and CIM Schema
- `http://www.dmtf.org/standards/cim`



fedora

CIM Specification

- Describes the language, naming, Meta Schema (formal definition of the model - terms used to express the model and their usage and semantics)
- Basic elements of the Meta Schema are Classes, Properties and Methods (also supports Indications and Associations as types of Classes and References as types of Properties, Qualifiers, Instances, etc.)
- Defines the details for integration with other management models (e.g. SNMP MIBs)
- Model can be expressed graphically in UML (Unified Modelling Language) or textually using a language called "mof" (Managed Object Format)

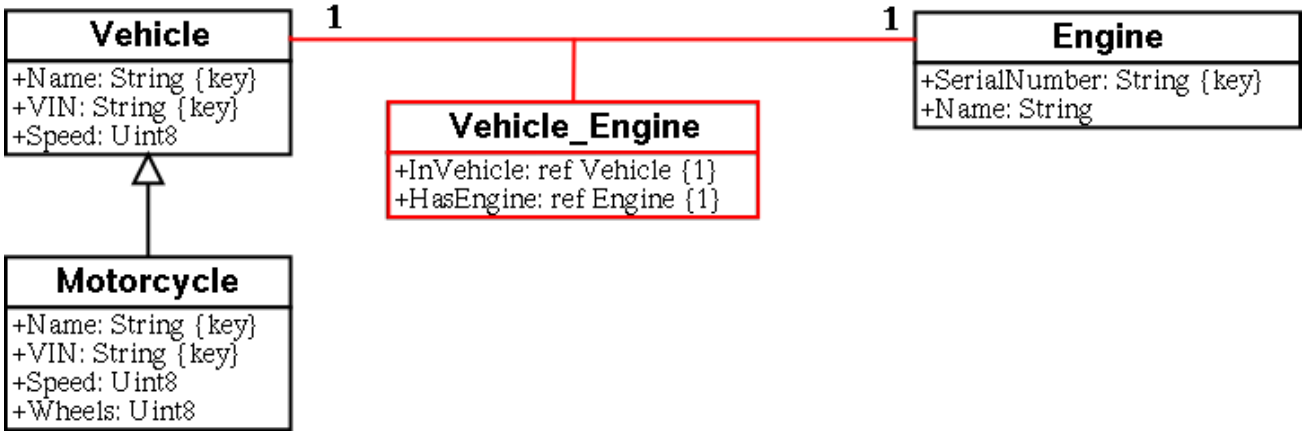
fedora

CIM Schema

- Provides the actual model descriptions
- Core Schema (essential classes) and Common Schema (important classes for various applications - storage, networking, desktop computing, etc.)
- DMTF simultaneously publish both an "Experimental" and a "Final" version of the schema
- Backward compatible (within same major version), can be easily expanded

fedora

Model Examle - UML/MOF




```
class Vehicle {
      [Key, Description (
          "The vendor's name of the vehicle")]
   String Name;
      [Key, Description (
          "The Vehicle Identification Number")]
   String VIN;
      [Description (
          "MPH of the vehicle")]
   Uint8 Speed;
};

class Motorcycle : Vehicle {
      [Description (
          "Number of wheels")]
   Uint8 Wheels;
};
```

fedora

Model Example - MOF

```
instance of Vehicle {
   Name="Ford Mustang GT";
   VIN="1FAFP90S45Y400167";
};

class Engine {
      [Key, Description (
          "Serial number of the engine")]
   String SerialNumber;
      [Description (
          "Name of engine ")]
   String Name;
};

instance of Engine {
  SerialNumber = "123902A323";
  Name = "8 CYLINDERS 5.4 Liters";
};

   [Association]
class Vehicle_Engine {
      [Key, Min ( "1" ), Max ( "1" ), Description (
          "The reference to the vehicle.")]
   Vehicle REF InVehicle;
      [Key, Min ( "1" ), Max ( "1" ), Description (
          "The reference to the engine.")]
   Engine REF HasEngine;
};

instance of Vehicle_Engine {
   InVehicle="Vehicle.Name=\"Ford Mustang GT\",VIN=\"1FAFP90S45Y400167\"";
   HasEngine = "Engine.SerialNumber=\"123902A323\"";
};
```

fedora

Object Path

- Unique identifier for instances and classes
- Combination of namespace, class name (and the values of all keys in case of instances)
- `objectPath = <namespacePath>:<modelPath>`
- namespacePath
  - namespaceType - protocol or API and address (e.g. http://10.34.24.224)
  - namespaceHandle - e.g. "root/cimv2" (it's not hiearchical, it's just one word)
- `modelPath = <class>.<key>=<value>,...,<key>=<value>`

fedora

Object Path - examples

- Full name of Vehicle instance from previous example:
  `'http://localhost/root/sample:`
  `Vehicle.Name="Ford Mustang GT",`
  `VIN="1FAFP90S45Y400167"'`
- Full name of Linux_UnixProcess instance might therefore be:
  `'http://localhost/root/cimv2:Linux_UnixProcess.`
  `CreationClassName="Linux_UnixProcess",`
  `CSCreationClassName="Linux_ComputerSystem",`
  `CSName="azrael2",Handle="2797",`
  `OSCreationClassName="Linux_OperatingSystem",`
  `OSName="azrael2"'`
- Full name of Linux_ComputerSystem class might be:
  `http://localhost/root/cimv2:Linux_ComputerSystem`
  (just `<key>=<value>` clauses are omitted)

fedora

CIM/XML encoding specification

- Specifies maping of CIM to XML
- No information is lost, vendors can distribute classes as XML

Model Example - XML

```
<INSTANCE CLASSNAME="Vehicle">
  <PROPERTY NAME="Speed" CLASSORIGIN="Vehicle" PROPAGATED="true" TYPE="uint8">
    <QUALIFIER NAME="Description" TYPE="string" TRANSLATABLE="true">
      <VALUE>MPH of the vehicle</VALUE>
    </QUALIFIER>
  </PROPERTY>
  <PROPERTY NAME="Name" CLASSORIGIN="Vehicle" TYPE="string">
    <QUALIFIER NAME="Key" TYPE="boolean" OVERRIDABLE="false">
      <VALUE>TRUE</VALUE>
    </QUALIFIER>
    <QUALIFIER NAME="Description" TYPE="string" TRANSLATABLE="true">
      <VALUE>The vendor&amp;apos;s name of the vehicle</VALUE>
    </QUALIFIER>
      <VALUE>Ford Mustang GT</VALUE>
  </PROPERTY>
  <PROPERTY NAME="VIN" CLASSORIGIN="Vehicle" TYPE="string">
    <QUALIFIER NAME="Key" TYPE="boolean" OVERRIDABLE="false">
      <VALUE>TRUE</VALUE>
    </QUALIFIER>
    <QUALIFIER NAME="Description" TYPE="string" TRANSLATABLE="true">
      <VALUE>The Vehicle Identification Number</VALUE>
    </QUALIFIER>
    <VALUE>1FAFP90S45Y400167</VALUE>
  </PROPERTY>
</INSTANCE>
```

fedora

CIM Operations over HTTP

- Defines what a management application (CIM client) can do with CIM, list of operations that a client might wish to perform
- Describes operations on classes, instances, qualifiers and associations
- Basicaly two kinds of methods:
  - Intrinsic methods - designed to be built into the CIM server, oriented towards manipulation of the model itself (retrieve, delete, create, enumerate, . . . generally manipulate classes, instances, associations and qualifiers)
  - Extrinsic methods - operations carried out by a method provider, which may do anything (shut the system down, bring it up, perform any complex operation)

fedora

Intrinsic methods

- Class-Oriented:
  - GetClass, DeleteClass, EnumerateClasses, EnumerateClassNames, (GetClassDefinition)
  - CreateClass, ModifyClass
- Instance-Oriented:
  - GetInstance, CreateInstance, DeleteInstance, ModifyInstance, EnumerateInstances, EnumerateInstanceName, GetProperty, SetProperty
- GetProperty and SetProperty are redundant, since GetInstance and ModifyInstance both allow a subset of properties (including one) to be specified
- MethodX vs. MethodXNames - the former case returns whole object, the latter case return only names (objectPath)

fedora

Provider

- Provider is "driver", interface between abstract model and real hardware/software
- Associted with dynamic entities (static entities can be defined in the mof code)
- Usually dynamic link library, one per class/association
- Installed at specific path known to CIM Server
- One library module can implement more than one type of provider

fedora

Types of provider

- Method Providers - handle calls to extrinsic methods to instances of classes
- Instance Providers - handle operations with instances of particular classes
- Property Providers - handle getting and setting properties on an instance
- Association (Associator) Providers - handle associations between classes or instances
- Indication (Indicator) Providers - handle events and alarms raised in the managed system
- Query Providers - handle database-style queries

fedora

Provider/CIM Server Interface

- Provider Protocol Adaptor
- In early days defined by particular CIM Server implementation (C++, Java)
- Problem for small devices where it was necessary to write providers in C
- Native Provider Interface (NPI) emerged, superseded by CMPI
- PPA is usually plug-in component of CIM Server now and CMPI is supported

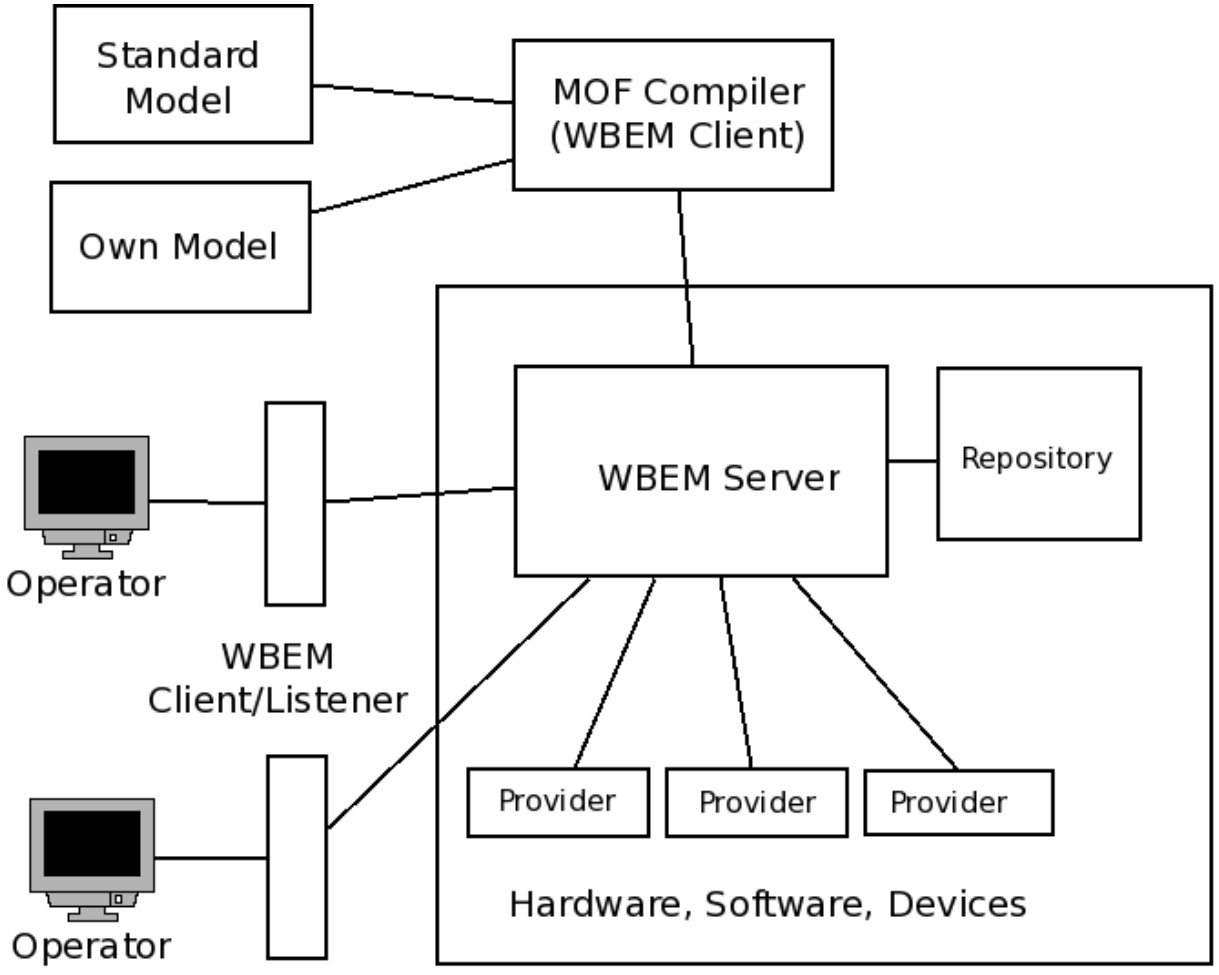
fedora

CMPI

- CMPI - Common Manageability Programming Interface, released by The Open Group
- Defines a programming interface between a CIM Server and providers
- In C, header files enabled for C++, C++ utility macros allows accessing the interface in C++ way
- Allows to write providers without having specific CIM Server libraries
- Provide support for remote providers
- Thread-safe, any number of providers in the same library module
- `http://www.opengroup.org/tech/management/cmpi/`

fedora

The WBEM Components - big picture



Part II

Availability in Fedora

fedora

Packages description, Examples, Questions

3 Packages description

- SBLIM Project
- Packages overview
- OpenPegasus
- Small Footprint CIM Broker
- WBEM Command Line Interface
- CIM Schema
- SBLIM Providers
- SBLIM Test Suite

4 Examples

5 Questions?

Section 3

Packages description

fedora

SBLIM Project

- SBLIM (pronounced "sublime") - Standards Based Instrumentation for Manageability, initiated by IBM, is is an umbrella project for a collection of Open Source systems management tools to enable WBEM on Linux
- CIMOMs are brokers, without any providers they have limited functionality, SBLIM brings mainly these providers (but also additional stuff)
- SBLIM providers instrument:
  - Operating system, processes
  - File systems, Network
  - NFSv3, NFSv4, Syslog
  - Kernel parameters, SysFS
- But also adds additional tools:
  - SBLIM TestSuite
  - WBEM client/server
- `http://sourceforge.net/projects/sblim/`

fedora

Packages overview

- CIM servers
  - OpenPegasus - tog-pegasus
  - Small Footprint CIM Broker - sblim-sfcb
- CIM clients
  - WBEM Command Line Interface - sblim-wbemcli
  - Small Footprint CIM Client - sblim-sfcc
  - CIM Client for Java - sblim-cim-client, sblim-cim-client2
- Providers
  - Base OS Instrumentation - sblim-cmpi-base
  - Filesystem and Volume Management Instrumentation - sblim-cmpi-fsvol
  - Network Instrumentation - sblim-cmpi-network
  - NFS Instrumentation - sblim-cmpi-nfsv3,4
  - Parameter Instrumentation - sblim-cmpi-params
  - RPM Instrumentation - sblim-cmpi-rpm
  - Sysfs Instrumentation - sblim-cmpi-sysfs
  - Syslog Instrumentation - sblim-cmpi-syslog
  - SMI-S standards based HBA CMPI Providers - sblim-smis-hba

fedora

Packages overview

- Support and development
  - SBLIM Test Suite - sblim-testsuite
  - Performance data gatherer - sblim-gather
  - CIM Schema - cim-schema
  - SBLIM CMPI Provider Development Support - sblim-cmpi-devel
  - Indication Helper - sblim-indication_helper
- WBEM System Management (WBEM-SMT)
  - Common functionality required by the task-specific resource access layers of WBEM-SMT - sblim-tools-libra
  - WBEM-SMT DNS task - sblim-cmpi-dns
  - WBEM-SMT Samba task - sblim-cmpi-samba
  - WBEM-SMT DHCP task - sblim-cmpi-dhcp

fedora

OpenPegasus (tog-pegasus)

- CIM server developed by The Open Group (vendor-neutral, technology-neutral consortium)
- Written in C++, under MIT license, portable
- Released twice a year, parallel versions (currently 2.8.X, 2.9.X, 2.10.X, 2.11.X), formal development (PEPs - Project Enhancement Proposals)
- A lot of features, but eats resources - not very suitable for e.g. embedded devices
- `http://www.openpegasus.org/`

fedora

OpenPegasus - basic commands

```
# service tog-pegasus start/stop/...
# cimserver
```

- tog-pegasus service, for the first time certificates are generated when starting with init script, server listens on port 5988 (http) or 5989 (https)

```
# osinfo
```

- prints information regarding the running operating system - good for quick test, whether the service runs properly (OperatingSystemModule must be enabled)

```
# cimconfig
```

- manages tog-pegasus configuration properties

```
# cimprovider
```

- manages registered CIM providers or CIM provider modules

fedora

OpenPegasus - basic commands

```
# wbemexec
```

- simple CIM Client, can submit CIM requests encoded in XML to a CIM Server

```
# cimmof
# cimmofl
```

- two versions of mof compiler, loads contetnt of mof file into the repository, former command is CIM Client and passes compiled output to CIM Server, the latter writes it directly into the repository (no need of running CIM Server, but more dangerous and meant mainly as debugging tool)

  - repository is placed in `/var/lib/Pegasus/repository/`

fedora

Small Footprint CIM Broker (sblim-sfcb)

- CIM server for resource-constrained and embedded environments
- Written in C, designed to be modular and lightweight
- Useful for providers debugging:
  - SBLIM_TRACE=[0..4] - level of trace (0 - no trace messages, 4 - all messages)
  - SBLIM_TRACE_FILE=/path/to/file - saving trace messages to the file instead of printing them to STDERR

fedora

WBEM Command Line Interface (sblim-wbemcli)

- Command line CIM Client
- Important options:
  - "-nl" - starts a new line for every property returned
  - "-dx" - shows XML communication between client and server
  - "-t" - adds array ([]), reference (&) and key property (#) indicators to property names
- ./wbemcli.ind file (or any file specified via WBEMCLI_IND) - can hold scheme and host specification to reduce typing
- `myCimom: http://root:password@localhost:5988`

Usage example:

```
wbemcli ei -nl myCimom/root/cimv2:CIM_OperatingSystem
```

fedora

CIM Schema (cim-schema)

- cim-schema (MOF files)
- cim-schema-docs (HTML schema description)
- Latest version 2.29.0, released 3 May 2011
- Required by sblim-sfcb, tog-pegasus ships schemata on his own

fedora

SBLIM Providers

- Provider is interface between CIM Server and real HW/SW
- Typical SBLIM provider consists from:
  - class/association specific libraries `/usr/lib/cmpi`
  - mof files and registration stuff `/usr/share/%{name}`
  - common libraries `/usr/lib`
  - documentation `/usr/share/doc/%{name}-%{version}`
- Many of them have also -test subpackage

fedora

SBLIM Test Suite (sblim-testsuite)

- Performing "hand-operated" tests of provider can become a nightmare - test suite was developed
- Allows to perform automated function verification tests against installed provider
- Aacts as CIM Client, uses sblim-wbemcli, used Perl and shell scripting
- Three types of tests:
  - Interface test - verifies proper implementation of all required provider interfaces
  - Consistence test - checks if provider returns meaningful values (compares them with script collected data)
  - Specification test - checks if the returned values follow the specification (compares them with meta-information about the model) - not implemented, moved to future;)
- `http://sblim.sourceforge.net/doc/SBLIMTestSuite.pdf`

Section 4

Examples

Examples

Section 5

Questions?

Questions?

The end.

Thanks for listening.