

November 15-18, 2010



Santa Clara Marriott
Santa Clara, CA

Talking WS-Man and CIM/XML

in your favorite programming language

Klaus Kämpf
Architect Systems Management
SUSE Linux



What developers say

CIM is hard

Documentation is sparse

Usage is complex

Tools are basic

**Management
Application**

CIM/XML

WS-Management

CIMOM

Provider

Resource

Managed System

Traditional approaches

(CIM/XML)

C/C++

sblim-sfcc

Java (jsr48)

sblim-cim-client2

```
try {
    final CloseableIterator<CIMObjectPath> iterator =
        client.enumerateInstanceNames(
            new CIMObjectPath("CIM_Process", "root/cimv2"));
    try {
        while (iterator.hasNext()) {
            final CIMObjectPath pathIter = iterator.next();
            System.out.println(pathIter.toString());
        }
    } finally {
        iterator.close();
    }
} catch (WBEMException e) {
    e.printStackTrace();
}
```

Removing the clutter

pywbem

(Python)

```
import pywbem
conn = pywbem.WBEMConnection('http://localhost')

names = conn.EnumerateInstanceNames('CIM_Process')

for name in names:
    print name

print '%d CIM_Process names' % len(instance_names)
```

ruby-sfcc

(Ruby)

```
require 'sfcc'
include Sfcc::Cim

client = Client.connect('http://localhost:5988')

op = ObjectPath.new("root/cimv2", "CIM_Process")

result = client.instance_names(op)

result.each do |name|
  puts "result: #{name}"
end
```

WS-Management

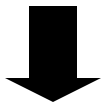
(openwsman)

Code generator (Swig)

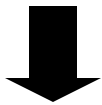
Focus on target language

Perl, Python, Ruby, Java, ...

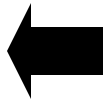
**Generic
API description**



SWIG



Generated bindings



**Specific
'syntactic sugar'**



Client code



```
XmlDoc result = client.enumerate(options, filter,
                                OpenWSManConstants.CIM_ALL_AVAILABLE_CLASSES);

if ((result == null) || result.isFault())
    System.err.println("Enumeration failed: "
        + ((result != null) ? result.fault().reason() : "?"));
else {
    String context = result.context();
    while (context != null) {
        result = client.pull(options, null,
                            OpenWSManConstants.CIM_ALL_AVAILABLE_CLASSES, context);
        if (result == null || result.isFault()) {
            System.err.println("Pull failed: "
                + ((result != null) ? result.fault().reason() : "?"));
            context = null;
            continue;
        }
        System.out.println(result.encode("UTF-8"));
        context = result.context();
    }
}
```

```
uri = Openwsman::XML_NS_CIM_INTRINSIC + "/CIM_Process"
result = client.enumerate( options, nil, uri )

loop do
  context = result.context
  break unless context
  result = client.pull( options, nil, uri, context )
  break unless result
  node = result.body.PullResponse.Items.child
  print node.Caption, node.Name
end
```

What developers want

Abstraction from protocol

Focus on resource model

ActiveRecord pattern

```
class Linux_EthernetPort < ActiveCim::Base
  self.site = "http://localhost/root/cimv2"
end
```

```
ports = Linux_EthernetPort.find(:all)
```

```
ports.each do |port|
  puts port.id
  port.device_id
  port.system_name
end
```

Summary

Lower the entry barrier
Abstract from the protocol
Focus on the resource
In any programming language

Thanks !

kkaempf@suse.de

References

CIM/XML

C/C++

<http://sblim.wiki.sourceforge.net> (sblim-sfcc)

Java

<http://sblim.wiki.sourceforge.net> (sblim-cim-client2)

Python

<http://pywbem.sourceforge.net/>

Ruby

<https://github.com/dmacvicar/ruby-sfcc>
<https://github.com/dmacvicar/activecim>

WS-Man

<http://www.openwsman.org>

**RPM
packages**

[https://build.opensuse.org/project/show?
project=systemsmanagement:wbem](https://build.opensuse.org/project/show?project=systemsmanagement:wbem)