

November 15-18, 2010



Santa Clara Marriott
Santa Clara, CA

Coding Providers

in your favorite programming language

Klaus Kämpf
Architect Systems Management
SUSE Linux



What developers say

CIM is hard

Documentation is sparse

Instrumentation is complex

Tools are simple

**Management
Application**



CIMOM

Provider

Resource

Managed System

Example

CMPI Provider

```
static CMPIStatus
EnumInstanceNames(
    CMPIInstanceMI * self,
    const CMPIContext * context,
    const CMPIResult * result,
    const CMPIObjectPath * reference)
{
    char * namespace =
        CMGetCharPtr(
            CMGetNameSpace(reference, NULL));
    ...
}
```

What developers want

Low entry barrier

Relevant use cases

Short path to success

More instrumentation

Resource model

Put the resource in focus

Hide boilerplate code

Use scripting language

Scripting

Ease Debugging

**Focus on solving the problem of
instrumentation**

Greatly reduced time to market

Less opportunity for error

Code generators

Facts

Python

Process Provider

- C++: 3600**
- CMPI: 3338**
- Python: 800**

File System Provider

- C++: 2900**
- CMPI: 1383 (subset of classes/features)**
- CIMPLE: 1230 + 8650 generated**
- Python: 440**

Concerns

about using a scripting language

One more tool

Speed

Footprint

Security

cmpi-bindings

Open source
CIMOM neutral
Support many languages
Natural language API
Code similarity

CIMOM

CMPI

CMPI Provider

cmapi-bindings

\$language adapter

Provider code

**Class
implementation**

Resource

Code Comparison

C vs Ruby

```
static CMPIStatus
EnumInstanceNames(
    CMPIInstanceMI * self,
    const CMPIContext * context,
    const CMPIResult * result,
    const CMPIObjectPath * reference)
{
    char * namespace =
        CMGetCharPtr(
            CMGetNameSpace(reference, NULL));
    ...
}
```

```
def enum_instance_names(context, result, reference)
  namespace = context.namespace
  ...
end
```

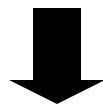
Generators

**Class definition
(MOF)**

Generator



Registration



**Provider
boilerplate**



**Class
template**



Provider

Python

pywbem

(+ YAWN)

Ruby

rubygem-genprovider

Summary

Instrumentation is easy

Focus on the resource

Optimize where needed

Open source drives adoption

Thanks !

`kkaempf@suse.de`

References

cmapi-bindings

<https://github.com/kkaempf/cmapi-bindings>

Python

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

<http://pywbem.wiki.sourceforge.net/Provider+QuickStart>

Ruby

<https://github.com/kkaempf/mof>

<https://github.com/kkaempf/genprovider>

RPM packages

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