Integração de Sistemas Legados com Plone

Integração de Sistemas Legados com Plone

Fabiano Weimar dos Santos [Xiru]

xiru@xiru.org

***II PyCon Brasil** - 2006*

Interlegis - Brasília - DF

O que iremos ver?

- O Problema
- Servidores de Web Services
  - Python + Twisted-web
  - Python + Zope + SOAPSupport
  - Java + Apache Tomcat + Apache Axis
- Clientes de Web Services
  - Python + SOAPpy
- Dicas

O que não iremos ver?

- Conectores de banco de dados relacional no Zope.
- Especificações de padrões de Web Services.

O Problema

- Integrar tecnologias diferentes **não é uma tarefa trivial**.
- O Plone costuma ser muito bem aceito como CMS, mas não tem tanta aceitação como plataforma de desenvolvimento de aplicações corporativas.
- Poucas empresas possuem uma infra- estrutura de TI uniforme. Mesmo quando isso acontece, sempre é necessário integrar tecnologias diferentes.
- Como integrar o Plone com aplicações legadas escritas em J2EE e .Net (ou mesmo COBOL)?

Web Services

- Service-oriented architecture (SOA)
  - http://en.wikipedia.org/wiki/Service-oriented_architectu
- Simple Object Access Protocol (SOAP)
  - http://www.w3.org/TR/soap/
- Web Services Description Language (WSDL)
  - http://www.w3.org/TR/wsdl
- Há muitos padrões de Web Services (OASIS)
  - http://www.oasis-open.org/specs/index.php

Vantagens

- Portabilidade entre diferentes plataformas de hardware e software.
- Interoperabilidade entre diferentes linguagens de programação.
- Encapsulamento da modelagem dos dados das aplicações legadas.
- O Plone se preocupa apenas com as funcionadades de CMS e repassa as funcionalidades de aplicações para servidores de aplicação corportativos.

Desvantagens

- Implementação não trivial
  - Contornável usando bons frameworks e ferramentas.
- Performance
  - Contornável usando boas técnicas de programação.
- Overhead

Servidor Python + Twisted-web

```
from twisted.web import soap, xmlrpc, resource, server
import os

def getQuote():
    return "PyCon Brasil II"

class XMLRPCQuoter(xmlrpc.XMLRPC):
    def xmlrpc_quote(self):
        return getQuote()

class SOAPQuoter(soap.SOAPPublisher):
    def soap_quote(self):
        return getQuote()
```

Servidor Python + Twisted-web

```
def main():
    from twisted.internet import reactor
    root = resource.Resource()
    root.putChild('RPC2', XMLRPCQuoter())
    root.putChild('SOAP', SOAPQuoter())
    reactor.listenTCP(7080, server.Site(root))
    reactor.run()

if __name__ == '__main__':
    main()
```

Cliente Python + SOAPpy

```
import xmlrpclib, SOAPpy

s = xmlrpclib.Server('http://localhost:7080/RPC2')
print "XML-RPC:", s.quote()

s = SOAPpy.SOAPProxy("http://localhost:7080/SOAP")
print "SOAP:", s.quote()
```

Servidor Java + Apache Tomcat + Apache Axis

```
public class TomcatAxis {

  public String getQuote() {
    return "PyCon Brasil II";
  }

}
```

Cliente Python + SOAPpy (WSDL)

```
from SOAPpy.WSDL import Proxy
url = 'http://localhost:8080/axis/TomcatAxis.jws?wsdl'
p = Proxy(url)
print p.getQuote()
```

Dicas

- Evite implementar Web Services que levam muito tempo para processar uma requisição.
  - Implemente “timeout” nas chamadas WSDL e SOAP.
- Arquivos binários devem ser encodados em Base64 para serem passados como parametros.
- Cuidado com métodos que potencialmente podem retornar grandes quantidades de dados (vetores).

Dicas

- Evite implementar métodos que recebam tipos complexos como parametros. O trabalho de implementar corretamente o marshalling de namespaces pode não compensar.
- Cuidado com chamadas de métodos que modificam dados. Para evitar problemas de integridade transacional, evite chamar varios métodos. Chame apenas um método que receba um documento XML como parametro.

Links

- Creating XML-RPC Servers and Clients with Twisted
  - http://tinyurl.com/6wl5r
- Twisted and WSDL
  - http://tinyurl.com/oo7xn
- Chapter 16. Web Services [DRAFT(1.0)] Part III. J2EE Applications on Geronimo
  - http://tinyurl.com/nb5fo

Links

- Which style of WSDL should I use?
  - http://tinyurl.com/b4uql
- PythonCLServiceTool: Python tools for exposing legacy applications as Web Services
  - http://dsd.lbl.gov/gtg/projects/PythonCLServiceTool/
- pyGridWare: Python Web Services Resource Framework
  - http://dsd.lbl.gov/gtg/projects/pyGridWare/index.html