UNIVERSIDADE FEDERAL DE OURO PRETO

Innan P. R. Amorim – 16.2.8416

Relatório RPC

João Monlevade
2018

Introdução

Nesta prática será apresentada a comunicação entre o cliente e o servidor, no qual o servidor realiza as operações básicas matemáticas e é retornada no cliente o resultado, utilizando a chamada de procedimento remoto(RPC), que permite aos clientes chamarem os procedimentos de forma transparentes em programas servidores que estejam executando em processos separados e em computadores de diferentes clientes.

Objetivos

A troca de mensagens entre o servidor e o cliente, através de um endereço local e portas de comunicação, através do RPC.

Desenvolvimento

O desenvolvimento foi realizado a partir da linguagem de programação Python. Criando um objeto para chamadas de método vago, o servidor é o alvo eventual da chamada. As chamadas podem ser feitas para o objeto do resultado, mas elas retornarão imediatamente null e armazenarão somente o nome e os parâmetros da chamada no objeto de multichamadas. Ao chamar o próprio objeto, isso faz com que todas as chamadas armazenadas sejam transmitidas como um único pedido system.multicall. O resultado desta chamada é um gerador, isto é, iterar sobre este gerador produz os resultados individuais.

- Servidor

```
from xmlrpc.server import SimpleXMLRPCServer
'''Funções responsáveis por realizar as operações aritiméticas de Soma,
Subtração, Multiplicação, divisão entre dois valores recebidos pelo cliente
e retornar o resultado da operação'''
def add(x,y):
    return x+y
def subtract(x, y):
    return x-y
def multiply(x, y):
    return x*y
def divide(x, y):
    return x/y

print("Trabalho de Sistemas Distribuidos - Professora Carla Lara.")

'''Inicialização do servidor na rede local e a porta de comunicação 8000'''
server = SimpleXMLRPCServer(("localhost", 8000))
print("Aguardando conexão na porta 8000")

'''Define que a entrada será multichamada, ou seja,
várias funções serão acionadas pelo cliente'''
server.register_multicall_functions()

'''Registro das funções de soma, subtração, multiplicação e divisão'''
server.register_function(add, 'add')
server.register_function(subtract, 'subtract')
server.register_function(multiply, 'multiply')
server.register_function(divide, 'divide')

'''Continua no loop do server'''
server.serve_forever()
#Referência: https://docs.python.org/2/library/xmlrpclib.html
```

Figura 1: Servidor

O servidor realiza as funções básicas matemáticas, soma, subtração, multiplicação e divisão. E retornam o valor do resultado no cliente. No servidor está definido que a entrada será de multichamadas, ou seja, várias funções serão acionadas pelos clientes.

- Cliente

```
import xmlrpc.client

a = int(input('Digite o primeiro número inteiro: '))
b = int(input('Digite o segundo número inteiro: '))

'''Inicia a conexão com o servidor no endereço
e porta definidos e informa que a conexão
será multichamadas'''
proxy = xmlrpc.client.ServerProxy("http://localhost:8000/")
multicall = xmlrpc.client.MultiCall(proxy)


'''Procedimento de chamada das funções aritiméticas'''
multicall.add(a,b)
multicall.subtract(a,b)
multicall.multiply(a,b)
multicall.divide(a,b)

'''Inicia a multichamada'''
result = multicall()

'''Exibição do resultado através da tupla no print abaixo'''
print("a+b=%d, a-b=%d, a*b=%d, a/b=%d" % tuple(result))
```

Figura 2: Cliente

O cliente inicia a conexão com o servidor pelo endereço e portas já definidos, informando que a comunicação será de multichamadas. Ele faz a chamada de cada função e o resultado é entregue a uma tupla.

- Resultado Servidor

```
C:/Users/range/PycharmProjects/Servidor/servidor.py
Trabalho de Sistemas Distribuídos - Professora Carla Lara.
Aguardando conexão na porta 8000
127.0.0.1 - - [27/Sep/2018 16:48:48] "POST / HTTP/1.1" 200 -
```

Figura 3: Teste Servidor

Teste do servidor para estabelecer a comunicação com o cliente.

- Resultado Cliente

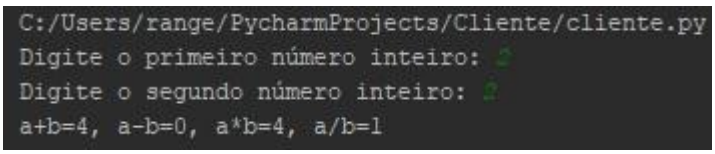

Figura 4: Teste Cliente

Teste do cliente após estabelecer a comunicação com o servidor, no qual foi necessário ter o privilégio do administrador para se utilizar os comandos no cmd.

Conclusão

O objetivo do RPC foi permitir ao programador se concentrar nas tarefas exigidas de um aplicativo, e ao mesmo tempo, tornando transparente para o programador o mecanismo que permite que as partes do aplicativo se comuniquem através de uma rede.

Referência

- https://docs.python.org/2/library/xmlrpclib.html