

Nama :

Aditya Alif Nugraha                      1301154183

Nadine Azhalia Purbani                      1301154519

Regita Anjani                                  1301154477

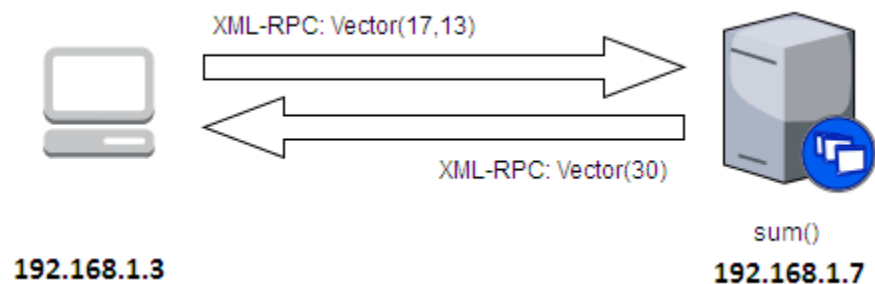
Rismada Gerra Nindya                      1301154561

Kelas : IF-39-01

## Tugas 2: Remote Procedure Call

### Definisi

RPC adalah memanggil procedure yang ada remote node. Contoh: terdapat sebuah klien yang akan memanggil prosedur ( bernama: *sum()* ) di suatu server. Setelah server memperoleh panggilan dari client kemudian server mengembalikan hasil penjumlahan kepada client.



Dengan menggunakan RPC, seolah-olah klien memanggil prosedur lokal padahal klien memanggil prosedur (*sum*) yang berada di server. Klien cukup mengetahui nama prosedur yang akan dipanggil dan memberikan parameter yang sesuai.

Terdapat banyak cara untuk implementasi RPC. Pada tugas ini kita akan mengimplementasikan RPC akan menggunakan XML-RPC (<https://en.wikipedia.org/wiki/XML-RPC>) . Secara ringkas, XML-RPC adalah “a simple, portable way to make remote procedure calls over HTTP.”

### Encode (marshall dan unmarshall)

Pada jaringan, XML-RPC diencode sebagai XML. Jenis RPC lainnya akan menggunakan metode encoding yang berbeda-beda.

<methodCall>

<methodName>sample.sumAndDifference</methodName>

```
<params>

<param><value><int>5</int></value></param>

<param><value><int>3</int></value></param>

</params>

</methodCall>
```

### **Tipe Data yang disupport XML-RPC**

`int`

A signed, 32-bit integer.

`string`

An ASCII string, which may contain NULL bytes. (Actually, several XML-RPC implementations support Unicode, thanks to the underlying features of XML.)

`boolean`

Either true or false.

`double`

A double-precision floating point number. (Accuracy may be limited in some implementations.)

`dateTime.iso8601`

A date and time. Unfortunately, since XML-RPC forbids the use of timezones, this is very nearly useless.

`base64`

Raw binary data of any length; encoded using Base64 on the wire. Very useful. (Some implementations don't like to receive zero bytes of data, though.)

`array`

An one-dimensional array of values. Individual values may be of any type.

`struct`

A collection of key-value pairs. The keys are strings; the values may be of any type.

### Kegunaan RPC

RPC dapat dikarakteristikan sebagai **transaction-oriented communication** dimana:

- Transaksi terdiri dari single-request dan single-response
- Transaksi diinisiasi ketika client mengirim request dan diterminasi oleh respon server

Dengan RPC, **low latency lebih diutamakan dari pada high throughput.**

Contoh RPC:

- Microsoft Exchange Server
- File replication service (FSR) di windows
- Active directory replication
- DCOM
- SQL (query)

### **Python XML-RPC**

XML-RPC Library yang digunakan pada python adalah xmlrpclib (pada python2.x) dan xmlrpc (pada python 3.x). Pastikan gunakan library yang sesuai dengan python yang telah diinstall!

### **Server RPC**

```

from xmlrpc.server import SimpleXMLRPCServer
from xmlrpc.server import SimpleXMLRPCRequestHandler

# Membatasi hanya path /RPC2, tidak bisa mengakses path yang lain
# Sebaiknya dibatasi, jika tidak dibatasi dapat menimbulkan celah keamanan
class RequestHandler(SimpleXMLRPCRequestHandler):
    rpc_paths = ('/RPC2',)

# Membuat Server
with SimpleXMLRPCServer(("localhost", 8000),
                        requestHandler=RequestHandler) as server:

    # Untuk mengetahui metode yang diregister
    server.register_introspection_functions()

    # Pada python terdapat fungsi build in pow() untuk memangkatkan suatu
    # bilangan dengan bilangan lainnya

    # Mendaftar fungsi pow() tersebut sebagai sebuah metode yang dapat
    # diakses dari client
    server.register_function(pow)

    # Cara lain register/mendaftarkan fungsi. Buat fungsinya terlebih dahulu
    # kemudian register ke server.
    # 'add' adalah metode yang dapat dipanggil oleh client
    def adder_function(x,y):
        return x + y
    server.register_function(adder_function, 'add')

    # Register sebagai instance; metode yang dipanggil adalah 'mul'
    class MyFuncs:
        def mul(self, x, y):
            return x * y

    server.register_instance(MyFuncs())

    # Jalan terus server
    server.serve_forever()

```

RPC server berjalan pada localhost (IP:127.0.0.1) pada port 8000. Server mempunyai 3 buah metode yaitu: pow(x,y), add(x,y) dan mul(x,y).

## Client

Setelah sever dijaankan, jalankan python secara interaktif. Tuliskan perintah ini satu per satu

```
>>> import xmlrpc
```

```
>>> print (s.power(2,5))
```

32

```
import xmlrpc.client
```

```
# client tidak langsung terhubung ke server tetapi melalui stub(proxy)
# banyak hal yang dilakukan oleh stub (proxy) sehingga memudahkan programmer
s = xmlrpc.client.ServerProxy('http://localhost:8000')
```

```
print(s.pow(2,3)) # Returns 2**3 = 8
print(s.add(2,3)) # Returns 5
print(s.mul(5,2)) # Returns 5*2 = 10
```

```
# Melihat semua metode yang disediakan server
print(s.system.listMethods())
```

## Tugas

1. Buatlah metode di server yang dapat menjumlah n bilangan.

Input: 2,3,7,5,2,4 (jumlah masukan bebas)

Output: 23

[illegible]

```

C:\Users\Rerre>"D:\Folder Kuliah\SEMESTER 6 (SEMANGAT)\SISTER\tugas_2_rpc-201802
14T033437Z-001\tugas_2_rpc\simple_client_rpc.py"
8
5
10
['add', 'mul', 'pow', 'sum', 'system.listMethods', 'system.methodHelp', 'system.
methodSignature']
3
31
C:\Users\Rerre>

```

## 2. Download tugas\_a\_rpc\_client.py dan tugas\_a\_rpc\_server.py

- Baca dan beri komentar setiap line pada source code. Apa yang dilakukan oleh program?

```

*D:\Folder Kuliah\SEMESTER 6 (SEMANGAT)\SISTER\tugas_2_rpc-20180214T033437Z-001\tugas_2_rpc\tugas_a_rpc_server.py - Notepad++
1 from xmlrpc.server import SimpleXMLRPCServer
2 import xmlrpc.client
3
4 def file_download():#membuat fungsi download
5     handle = open("aditya alif nugraha-foto.jpg","rb")#membuka file berupa .jpg yang dikirimkan server
6     return xmlrpc.client.Binary(handle.read())#mengembalikan hasil dalam bentuk biner
7     handle.close()#menutup koneksi host
8
9 server = SimpleXMLRPCServer(("10.20.3.242",8000))#membuat objek dari kelas SimpleXMLRPCServer dengan menggunakan IP server
10 print ("Listening on port 8000")#menampilkan pesan jika sudah terjadi koneksi
11
12 server.register_function(file_download,'download')#mendaftarkan fungsi yang dapat digunakan oleh client
13
14 server.serve_forever()#server akan terus menyala sampai PC dimatikan

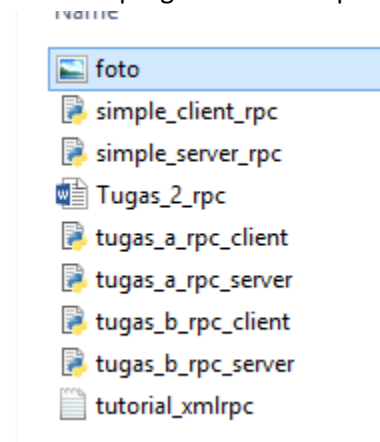
```

```

*D:\Folder Kuliah\SEMESTER 6 (SEMANGAT)\SISTER\tugas_2_rpc-20180214T033437Z-001\tugas_2_rpc\tugas_a_rpc_client.py
1 import xmlrpc
2 import xmlrpc.client
3
4 proxy = xmlrpc.client.ServerProxy('http://10.20.3.242:8000')#untuk mengenali IP dari server yang dituju
5 handle = open("foto.jpg","wb")#untuk membaca file berupa .jpg
6 handle.write(proxy.download().data)#untuk menuliskan file yang diterima dari server
7 handle.close()#menutup koneksi host
8
9

```

- Jalankan program tersebut pada 2 komputer. Screenshoot hasil kerja program!



File yang diterima berupa .jpg dengan nama foto.

## 3. Download tugas\_b\_rpc\_client.py dan tugas\_b\_rpc\_server.py

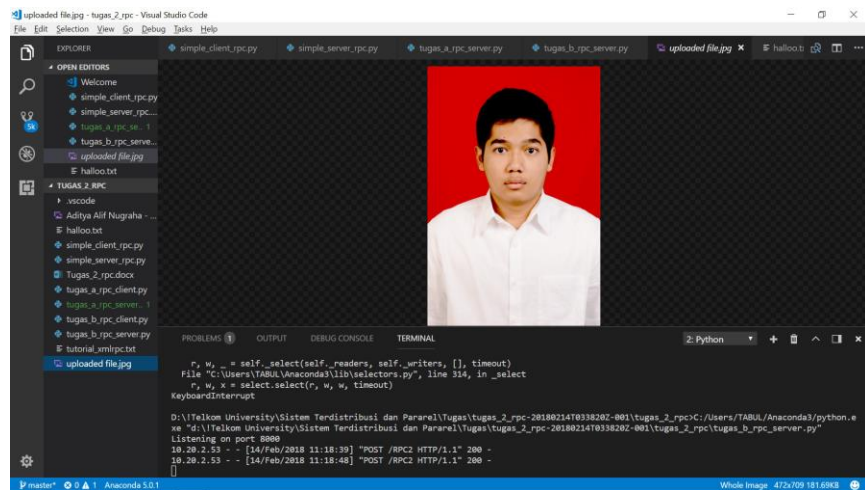
- Baca dan beri komentar setiap line pada source code. Apa yang dilakukan oleh program?

```
*D:\Folder Kuliah\SEMESTER 6 (SEMANGAT)\SISTER\tugas_2_rpc-20180214T033437Z-001\tugas_2_rpc\tugas_b_rpc_server.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
simple_server_rpc.py L3 tugas_a_rpc_client.py L3 tugas_a_rpc_server.py L3 tugas_b_rpc_client.py L3 tugas_b_rpc_server.py L3
1 from xmlrpc.server import SimpleXMLRPCServer # mengimport class SimpleXMLRPCServer dari modul xmlrpc.server
2
3 def file_upload(filedata): # membuat fungsi file upload dengan parameter file data yang akan dikirim dari client
4     with open("uploaded file.jpg", 'wb') as handle: # menentukan tempat untuk menyimpan file yang diupload dari client
5         data=filedata.data #convert from byte to binary IMPORTANT!
6         handle.write(data) # menulis data yang diupload dari client
7         handle.close() # menutup file yang telah ditulis
8         return True #IMPORTANT
9
10 # must have return value
11 # else error message: "cannot marshal None unless allow_none is enabled"
12
13 server = SimpleXMLRPCServer(("10.20.3.242",8000)) # membuat object baru dari class SimpleXMLRPCServer
14 print ("Listening on port 8000") # memberi informasi bahwa server siap menerima koneksi pada port 8000
15
16 server.register_function(file_upload,'file_upload') # mendaftarkan fungsi yang dapat digunakan oleh client
17
18 server.serve_forever() # membuat server online selamanya, kecuali dimatikan paksa
```

b.

```
*D:\Folder Kuliah\SEMESTER 6 (SEMANGAT)\SISTER\tugas_2_rpc-20180214T033437Z-001\tugas_2_rpc\tugas_b_rpc_client.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
simple_server_rpc.py L3 tugas_a_rpc_client.py L3 tugas_a_rpc_server.py L3 tugas_b_rpc_client.py L3
1 import xmlrpc.client
2
3 proxy = xmlrpc.client.ServerProxy("http://10.20.3.242:8000")#untuk mengenali IP dari server yang dituju
4
5 with open("foto.jpg", 'rb') as handle:#untuk membaca file berupa .jpg
6     data = xmlrpc.client.Binary(handle.read())#membuat objek untuk mengubah data menjadi biner
7     handle.close()#menutup koneksi host
8     result = proxy.file_upload(data)#membuat objek untuk mengirim data
```

c. Jalankan program tersebut pada 2 komputer. Screenshoot hasil kerja program!



4. Buatlah program ftp sederhana menggunakan source code yang telah diberikan













```
D:\Folder Kuliah\SEMESTER 6 (SEMANGAT)\SISTER\tugas_2_rpc-20180214T033437Z-001\tugas_2_rpc\tugas_4_server.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
tugas_b_rpc_client.py | tugas_b_rpc_server.py | tugas_4_client.py | tugas_4_server.py |
1 from xmlrpc.server import SimpleXMLRPCServer
2 import xmlrpc.client
3
4
5 def file_download():
6     handle = open("Aditya Alif Nugraha - Photo.jpg", 'rb')
7     return xmlrpc.client.Binary(handle.read())
8     handle.close()
9
10
11 # membuat fungsi file upload dengan parameter file data yang akan dikirim dari client
12 def file_upload(filedata):
13     # menentukan tempat untuk menyimpan file yang diupload dari client
14     with open("uploaded file.jpg", 'wb') as handle:
15         data1 = filedata.data # convert from byte to binary IMPORTANT!
16         handle.write(data1) # menulis data yang diupload dari client
17         handle.close() # menutup file yang telah ditulis
18         return True # IMPORTANT
19 # must have return value
20 # else error message: "cannot marshal None unless allow_none is enabled"
21
22
23 server = SimpleXMLRPCServer(('10.20.3.242', 8000))
24 print("Listening on port 8000")
25 server.register_introspection_functions()
26 server.register_function(file_download, 'download')
27 # mendaftarkan fungsi yang dapat digunakan oleh client
28 server.register_function(file_upload, 'file_upload')
29
30
31 server.serve_forever()
```

```
D:\Folder Kuliah\SEMESTER 6 (SEMANGAT)\SISTER\tugas_2_rpc-20180214T033437Z-001\tugas_2_rpc\tugas_4_client.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
tugas_b_rpc_client.py | tugas_b_rpc_server.py | tugas_4_client.py | tugas_4_server.py |
1 import xmlrpc.client
2
3 s = xmlrpc.client.ServerProxy('http://10.20.3.242:8000')
4 handle = open("foto.jpg", "wb") # untuk membaca file berupa .jpg
5 handle.write(s.download().data) # untuk menuliskan file yang diterima dari server
6 handle.close() # menutup koneksi host
7
8 # Print list of available methods
9 print(s.system.listMethods())
10
```

```
C:\Users\Berre>"D:\Folder Kuliah\SEMESTER 6 (SEMANGAT)\SISTER\tugas_2_rpc-20180214T033437Z-001\tugas_2_rpc\tugas_4_client.py"
['download', 'file_upload', 'system.listMethods', 'system.methodHelp', 'system.methodSignature']
C:\Users\Berre>
```

HASIL :

-  foto
-  simple\_client\_rpc
-  simple\_server\_rpc
-  Tugas\_2\_rpc
-  tugas\_4\_client
-  tugas\_a\_rpc\_client
-  tugas\_a\_rpc\_server
-  tugas\_b\_rpc\_client
-  tugas\_b\_rpc\_server
-  tutorial\_xmlrpc