Group -
Karan Shah(54)
Trusha Talati(60)
Soham Vaidya(63)

Experiment 1

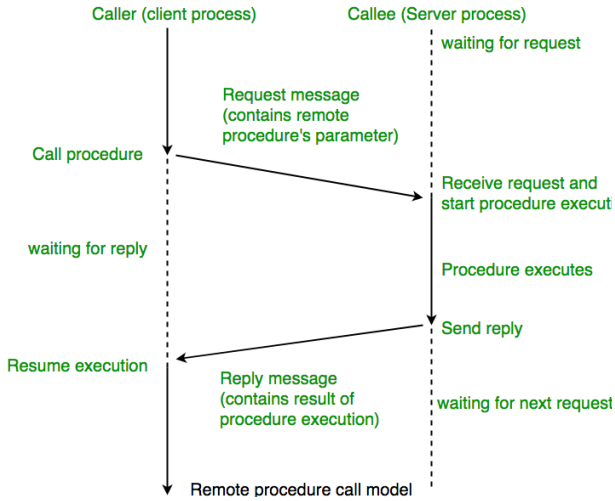
Aim: Client Server based program using RPC/RMI

Objectives: To understand how procedure calls like RPC and RMI work for constructing distributed, client-server based applications.

Theory:

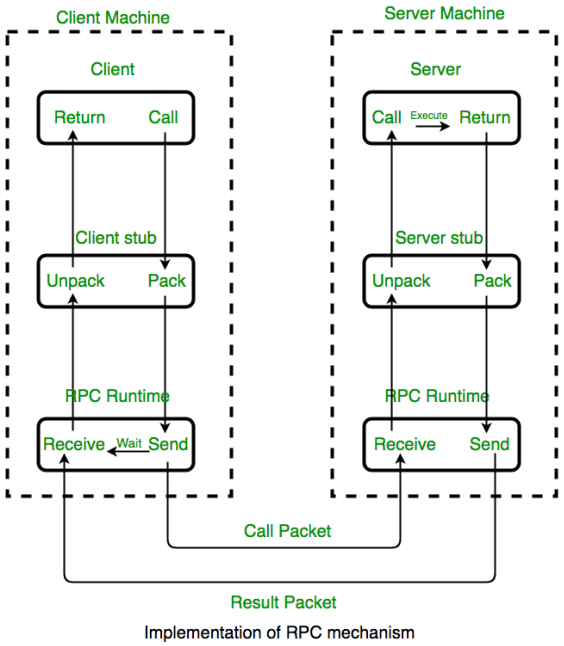
Remote Procedure Call (RPC) is a powerful technique for constructing distributed, client-server based applications. It is based on extending the conventional local procedure calling so that the called procedure need not exist in the same address space as the calling procedure. The two processes may be on the same system, or they may be on different systems with a network connecting them.

When making a Remote Procedure Call:



1. The calling environment is suspended, procedure parameters are transferred across the network to the environment where the procedure is to execute, and the procedure is executed there.

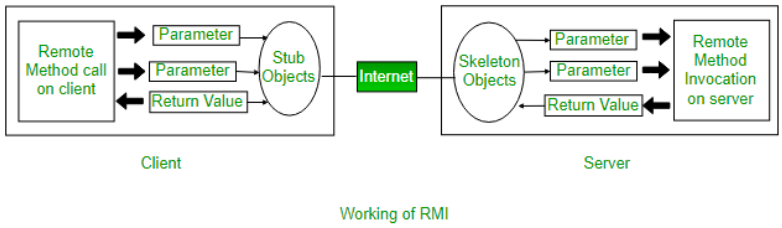
2. When the procedure finishes and produces its results, its results are transferred back to the calling environment, where execution resumes as if returning from a regular procedure call.



Implementation of RPC mechanism

Remote Method Invocation (RMI) is an API which allows an object to invoke a method on an object that exists in another address space, which could be on the same machine or on a remote machine.

Through RMI, object running in a JVM present on a computer (Client side) can invoke methods on an object present in another JVM (Server side). RMI creates a public remote server object that enables client and server side communications through simple method calls on the server object.



Working of RMI

XML-RPC :

XML-RPC is a Remote Procedure Call method that uses XML passed via HTTP as a transport. With it, a client can call methods with parameters on a remote server (the server is named by a URI) and get back structured data.

xmlrpc is a package that collects server and client modules implementing XML-RPC. The modules are:

- Xmlrpc.client
- xmlrpc.server

Code & Output:

server.py -

```
from xmlrpc.server import SimpleXMLRPCServer
import xmlrpc.client
import os

server = SimpleXMLRPCServer(('localhost',3000), allow_none=True)

prices = {'Phone': 20000, 'Bag': 2000, 'Cable': 300, 'TV': 30000, 'Chair':
2500}

def load_data():
    return prices
```

```
def calculate_total(cart):
    total = 0
    for k, v in cart.items():
        total += prices[k] * v
    return total

server.register_function(load_data)
server.register_function(calculate_total)


if __name__ == '__main__':
    try:
        print('Serving...')
        server.serve_forever()
    except KeyboardInterrupt:
        print('Exiting')
```

client.py -

```
from xmlrpc.client import ServerProxy

proxy = ServerProxy('http://localhost:3000')

cart = {}
items = {}
if __name__ == '__main__':
    prices = proxy.load_data()
    for i, key in enumerate(prices):
        items[i + 1] = key
    print("Enter your choices")
    while(True):
        for i, (k, v) in enumerate(prices.items()):
            print(i + 1, k + ": ", v)
        print(i + 2, "Find total and exit")
        choice = int(input())
        if choice == i + 2:
            break
        print("Enter the quantity")
        quantity = int(input())
```

```
        cart[items[choice]] = quantity
    print("Your total amount is", proxy.calculate_total(cart))
```

```
PS C:\Users\trusha\Desktop\Distributed-System-> & C:/Users/trusha/AppData/Local/Programs/Python/Python37-32/pyt
hon.exe c:/Users/trusha/Desktop/Distributed-System-/payment/server.py
Serving...
```

C:\windows\system32\cmd.exe

```
C:\Users\trusha\Desktop\Distributed-System-\payment>python client.py
Enter your choices
1 Phone:  20000
2 Bag:  2000
3 Cable:  300
4 TV:  30000
5 Chair:  2500
6 Find total and exit
1
Enter the quantity
1
1 Phone:  20000
2 Bag:  2000
3 Cable:  300
4 TV:  30000
5 Chair:  2500
6 Find total and exit
2
Enter the quantity
2
1 Phone:  20000
2 Bag:  2000
3 Cable:  300
4 TV:  30000
5 Chair:  2500
6 Find total and exit
6
Your total amount is 24000

C:\Users\trusha\Desktop\Distributed-System-\payment>
```

Conclusion:

In this experiment we explored various functions of Python's package for distributed computing - xmlrpc. We learnt use of its various functions and features to perform different operations on client and server.

References:

1. https://docs.python.org/3/library/xmlrpc.server.html
2. https://www.geeksforgeeks.org/remote-procedure-call-rpc-in-operating-system/