



Bài 3



1. RPC LÀ GÌ?

RPC (Remote Procedure Call) là cơ chế cho phép một chương trình gọi hàm ở máy khác như gọi hàm cục bộ.

So với message queue:

- RPC: request → response (đồng bộ)
- MQ: fire-and-forget / async

2. NGOÀI XMLRPC CÒN CÁC THƯ VIỆN RPC NÀO?

- Ngoài XML-RPC, còn rất nhiều thư viện RPC khác:
- JSON-RPC
- gRPC
- Apache Thrift
- SOAP
- REST (giả RPC)

=> JSON-RPC là lựa chọn phù hợp nhất để demo cho bài này do nhẹ, dễ demo, phù hợp bài học

3. JSON-RPC LÀ GÌ?

JSON-RPC là giao thức RPC:

- Dùng JSON làm định dạng dữ liệu
- Hoạt động trên HTTP hoặc TCP
- Đơn giản, dễ cài, dễ hiểu

🚩 Cấu trúc request:

```
json
{
  "jsonrpc": "2.0",
  "method": "add",
  "params": [3, 5],
  "id": 1
}
```

🚩 Response:

```
json
{
  "jsonrpc": "2.0",
  "result": 8,
  "id": 1
}
```

4. JSON-RPC DÙNG TRONG DEMO

Thư viện:

- jsonrpcserver (server)
- jsonrpcclient (client)
- Cài đặt:

```
pip install jsonrpcserver  
jsonrpcclient flask
```

Mô hình:

Client——>RPC(JSON/HTTP)——> Server

5. CODE SERVER VÀ CODE CLIENT

- Server

```
server.py 2 X client.py 1
server.py > ...
1 from jsonrpcserver import method, dispatch, Success
2 from flask import Flask, request
3
4 app = Flask(__name__)
5
6 @method
7 def add(a, b):
8     return Success(a + b)
9
10 @method
11 def multiply(a, b):
12     return Success(a * b)
13
14 @app.route("/rpc", methods=["POST"])
15 def rpc():
16     return dispatch(request.get_data().decode())
17
18 if __name__ == "__main__":
19     app.run(port=5000)
```

- Client

```
server.py 2 client.py 1 X
client.py > ...
4 def call_rpc(method, params):
8     payload = {
9         "method": method,
10        "params": params,
11        "id": 1
12    }
13
14    response = requests.post(
15        "http://localhost:5000/rpc",
16        json=payload
17    )
18
19    data = response.json()
20
21    if "error" in data:
22        print("RPC Error:", data["error"])
23    else:
24        print(f"Result {method}:", data["result"])
25
26 call_rpc("add", [4, 6])
27 call_rpc("multiply", [3, 5])
```

5. CHẠY VÀ KẾT QUẢ DEMO

- Terminal 1 – server

```
C:\Users\Minh\Downloads\UDPT\B3>python server.py
* Serving Flask app 'server'
* Debug mode: off
WARNING: This is a development server. Do not use
production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
█
```

```
C:\Users\Minh\Downloads\UDPT\B3>python server.py
* Serving Flask app 'server'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production
production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [01/Jan/2026 20:20:14] "POST /rpc HTTP/1.1" 200 -
127.0.0.1 - - [01/Jan/2026 20:20:16] "POST /rpc HTTP/1.1" 200 -
█
```

- Terminal 2 – client

```
C:\Users\Minh\Downloads\UDPT\B3>python client.py
Result add: 10
Result multiply: 15
```

5. Ý NGHĨA RPC TRONG HỆ PHÂN TÁN

- Client không cần biết server xử lý thế nào
- Giao tiếp chuẩn hóa bằng JSON
- Phù hợp microservices đơn giản