**PROBLEM #1**
There exist two servers that provide the following services for clients:
**Server #1:**
int Service1(int, int, int)
float Service1(float, int)
int Service2(int, int)
void Service4 (int L[], int N, int SL[]) // N and list L[] are input parameters and SL[] is an output parameter

**Server #2:**
float Service1(float, int)
float Service2(float, int)
int Service2(int, int)
int Service3(int, int)
void Service4 (int L[], int N, int SL[]) // N and list L[] are input parameters and SL[] is an output parameter

In addition, there are two clients: *Client_A* and *Client_B*
***Client_A*** may request the following services:
float Service1(float, int)
int Service3(int, int)
void Service4 (int L[], int N, int SL[])

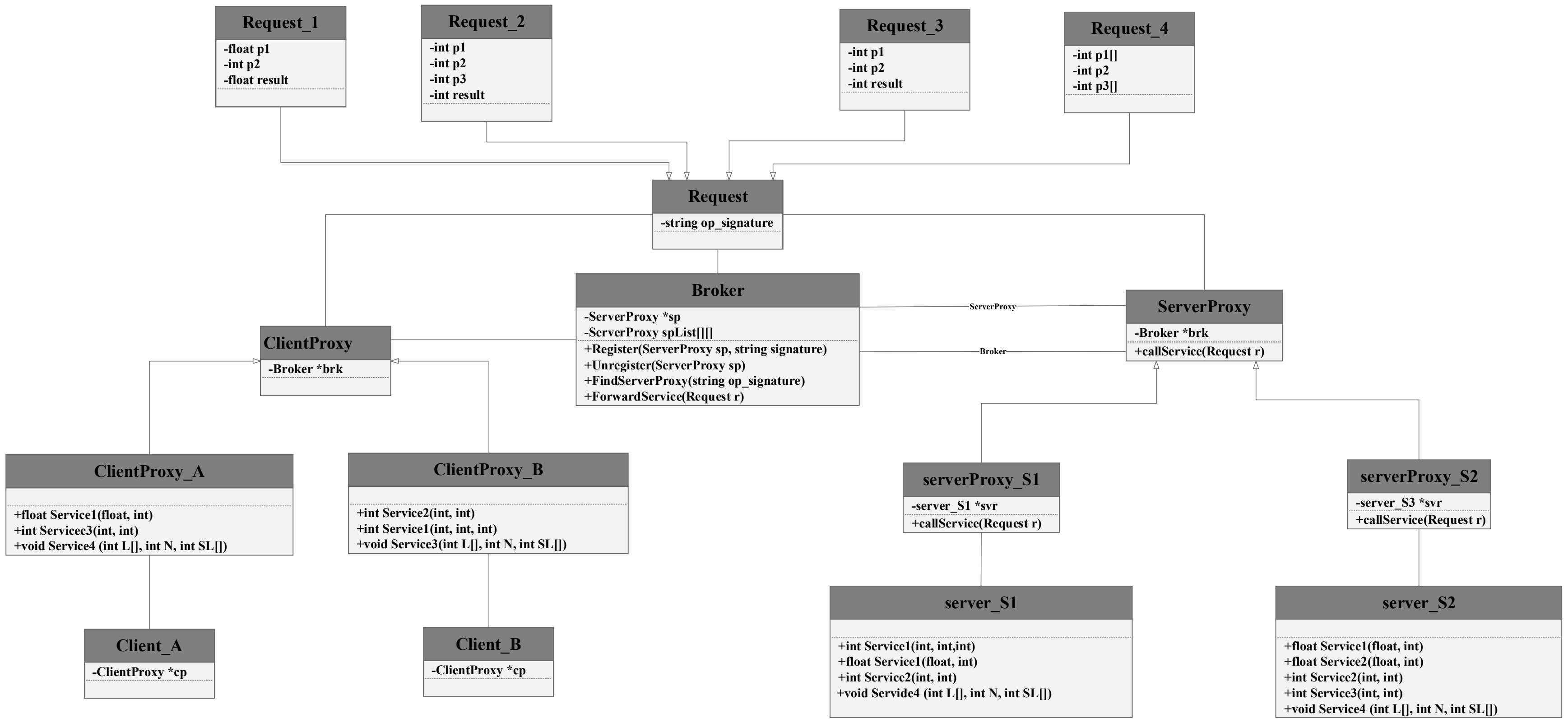
***Client_B*** may request the following services:
int Service2(int, int)
int Service1(int, int, int)
void Service4 (int L[], int N, int SL[])

The client processes do not know the location (pointer) to servers that may provide these services. Devise a software architecture using a **Client-Broker-Server** architecture for this problem. In this design the client processes are not aware of the location of servers providing these services.

- Provide a class diagram for the proposed architecture. In your design, all components should be **decoupled** as much as possible.
- Provide the **pseudocode** for all operations of the following components/classes:
    - Broker
    - Client Proxy of *Client-A*
    - Server Proxy of *Server-1.*
- Provide a sequence diagram to show how *Client_B* gets *Service4(int L[], int N, int SL[])* service.

-                    -

Class Diagram:



**Class ClientProxy_A**

Operations

```
float Service1(float x, int y) {
        r = new Request_1
        r->p1 = x
        r->p2 = y

        r->op_signature = “float Service1(float,int)”
        brk->ForwardService(r)

        return r->result
}

int Service3(int x, int y) {
        r = new Request_3
        r->p1 = x
        r->p2 = y


        r->op_signature = “int Service3(int,int)
        brk->ForwardService(r)

        return r->result
}

void Service4 (int L[], int N, int SL[]){
        r = new Request_4
        r->p1[] = L[]
        r->p2 = N

        r->op_signature = “void Service4(int[],int,int[])”
        brk->ForwardService(r)

        SL[] = r->p3[]
}
```

## Class Broker

```
ServerProxy *sp
ServerProxy spList[][]
Register(ServerProxy sp, String signature){
        add sp to spList
        add signature to spList[sp]
}

Unregister(ServerProxy sp){
        remove sp from spList
}

FindServerProxy(string op_signature){
        for every svrproxy in spList
            IF spList[svrproxy] contains op_signature THEN
                return svrproxy
            ENDIF
}

ForwardService(Request r){
        sp = FindServerProxy(r->op_signature)
        IF sp != null THEN
            sp->callservice(r)
        ENDIF
}
```

## Class ServerProxy_S1

Operations

```
callService(Request r){
        switch (r->op_signature)
            case "int Service1(int,int,int)"

                r->result = svr->Service1(r->p1, r->p2, r->p3)

            case "float Service1(float, int)":

                r->result = svr->Service1(r->p1, r->p2)

            case "int Service2(int,int)":

                r->result = svr->Service2(r->p1, r->p2)

            case "void Service4 (int[],int,int[])":

                svr->Service4(r->p1[], r->p2, r->p3[])
}
```

**Class ServerProxy**

Broker *brk

Operations

**callService(Request r) is an abstract operation**

**Class Request**

string op_signature

**Class Request 1**

| | |
|---|---|
| float p1 | //parameter 1 |
| int p2 | //parameter 2 |
| float result | //return value of the service |

**Class Request 2**

| | |
|---|---|
| int p1 | //parameter 1 |
| int p2 | //parameter 2 |
| int p3 | //parameter 3 |
| int result | //return value of the service |

**Class Request 3**

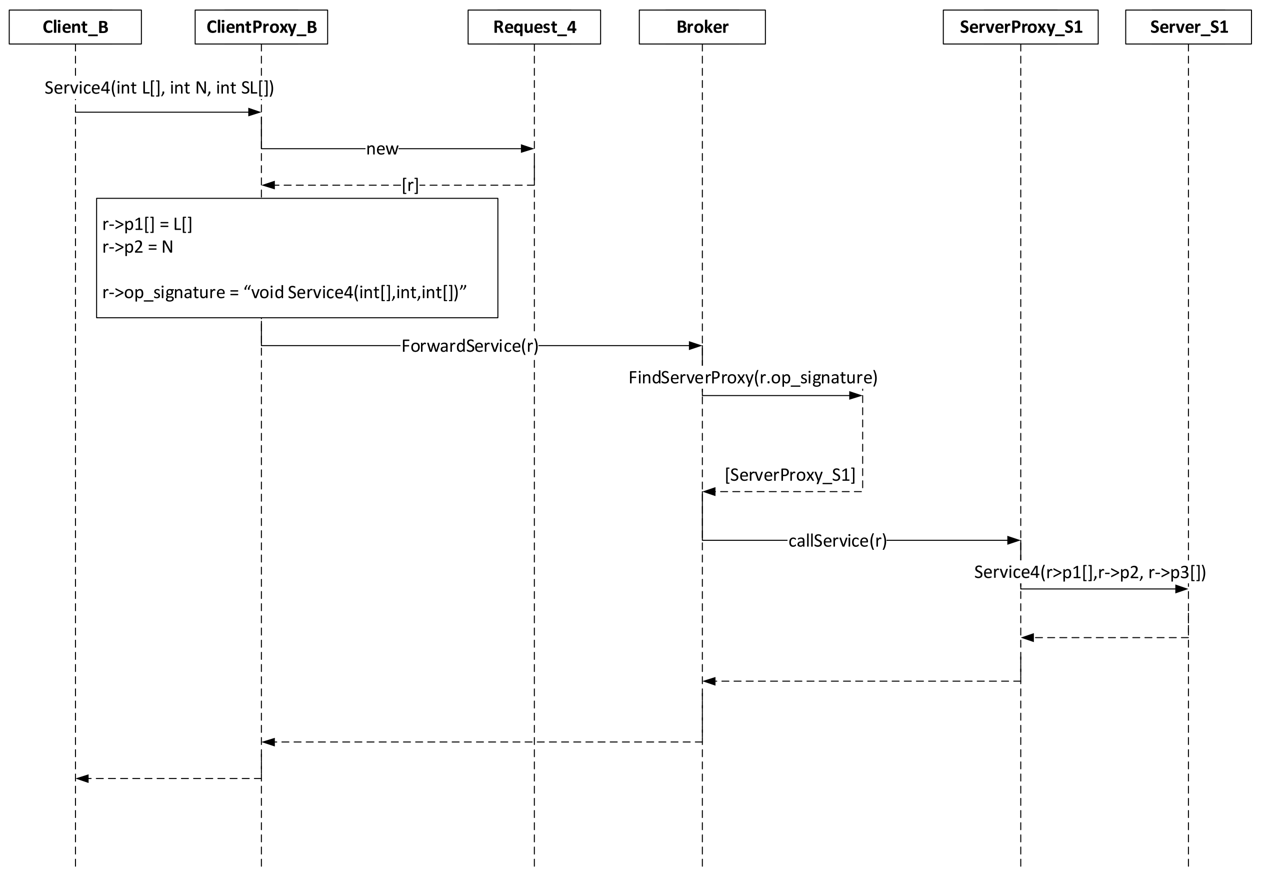
| | |
|---|---|
| int p1 | //parameter 1 |
| int p2 | //parameter 2 |
| int result | //return value of the service |

**Class Request 4**

| | |
|---|---|
| int p1[] | // parameter 1 |
| int p2 | // parameter 2 |
| int p3[] | //contains the result of the service |

Sequence Diagram:

Client A gets "Service4(int L[], int N, int SL[])" service


Client_B
ClientProxy_B
Request_4
Broker
ServerProxy_S1
Server_S1
Service4(int L[], int N, int SL[])
new
[r]
r->p1[] = L[]
r->p2 = N
r->op_signature = "void Service4(int[],int,int[])"
ForwardService(r)
FindServerProxy(r.op_signature)
[ServerProxy_S1]
callService(r)
Service4(r>p1[],r->p2, r->p3[])