Captain Kot

Michael Schick, Marcel Hasselberg,
Tizian Grossmann

Agenda

1

Architektur

2

Live Demo

3

Umsetzung

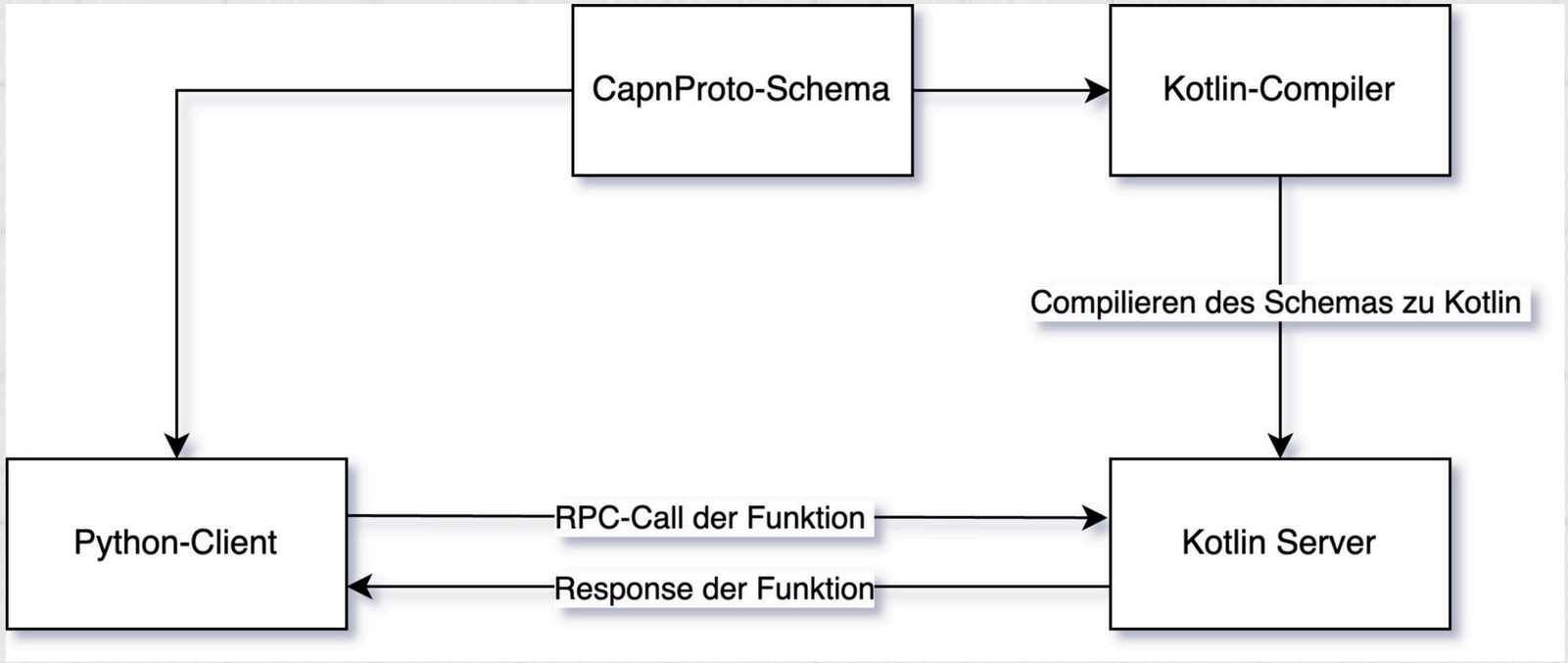
4

Fazit

1. Architektur

~Michael

Architektur



2.1 Live Demo Compiler

~Michael

2.2 Live Demo Gesamtsystem

~Michael

3. Umsetzung

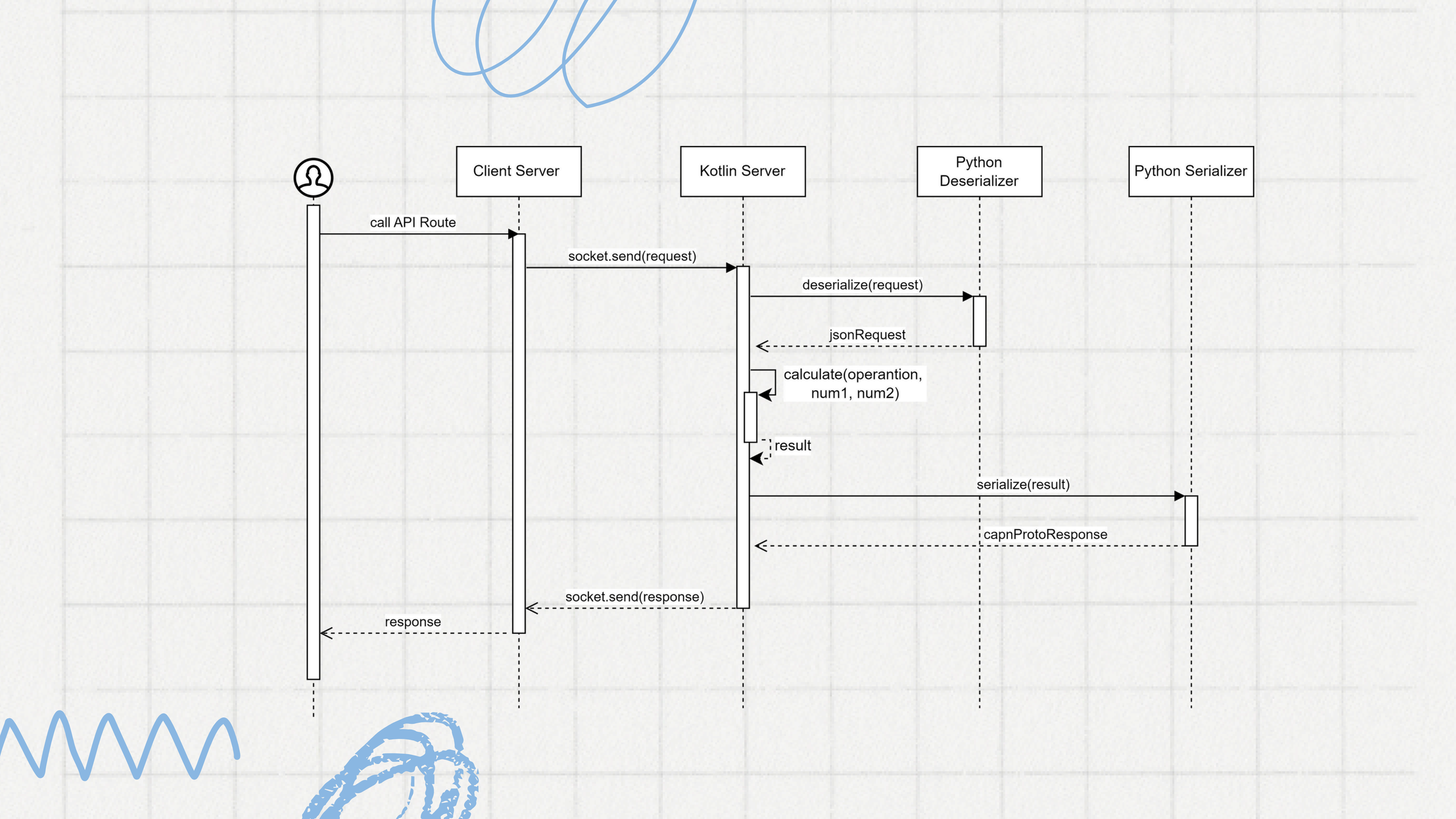
~Marcel

Server: Docker Multi Stage Build

-> Erst Schema Compilen

-> Kotlin Code in .jar Datei bringen

-> .jar ausführen


Client Server
Kotlin Server
Python Deserializer
Python Serializer
call API Route
socket.send(request)
deserialize(request)
jsonRequest
calculate(operantion, num1, num2)
result
serialize(result)
capnProtoResponse
socket.send(response)
response

Server: Deserialisierung der Request

Kotlin

```kotlin
// Start the Python process for deserialization
val deserializeProcess = ProcessBuilder("python3", "./src/main/resources/deserializer.py").start()

// Write the Client request to the Python process which deserializes it to JSON
deserializeProcess.outputStream.use { it.write(messageBytes) }
deserializeProcess.outputStream.close()

val deserializeErrorStream = deserializeProcess.errorStream.bufferedReader().readText()
val deserializeExitCode = deserializeProcess.waitFor()

if (deserializeExitCode == 0) {
    // If successful, read the response
    val responseBytes = deserializeProcess.inputStream.readBytes()
    val jsonResponse = JSONObject(String(responseBytes))
    println("jsonResponse: $jsonResponse")

    val operation = jsonResponse.getString("operation")
    val num1 = jsonResponse.getDouble("num1")
    val num2 = jsonResponse.getDouble("num2")

    val result = performCalculation(operation, num1, num2)
```

python

```python
capnpCalculatorSchema = capnp.load('calculator.capnp')

def deserializeRequest(requestData):
    with capnpCalculatorSchema.Request.from_bytes(requestData) as request:
        operation = request.operation
        num1 = request.num1
        num2 = request.num2

        response = {
            "operation": operation,
            "num1": num1,
            "num2": num2
        }

        response_json = json.dumps(response)
        response_bytes = response_json.encode('utf-8')

        # Send the deserialized request back to the kotlin server in json format
        sys.stdout.buffer.write(response_bytes)
        sys.stdout.buffer.flush()
```

Server: Serialisierung der Response

Kotlin

```kotlin
val serializeProcess = ProcessBuilder("python3", "./src/main/resources/serializer.py").start()

// Write the result to the serializer
serializeProcess.outputStream.use { it.write(result.toString().toByteArray()) }
serializeProcess.outputStream.close()

val serializeErrorStream = serializeProcess.errorStream.bufferedReader().readText()
val serializeExitCode = serializeProcess.waitFor()

if (serializeExitCode == 0) {
        // Read and send the final response
        val finalResponseBytes = serializeProcess.inputStream.readBytes()

        // Send the serialized response to the client
        outputStream.writeInt(finalResponseBytes.size)
        outputStream.write(finalResponseBytes)
        outputStream.flush()
} else {
    println("Error while serializing the response with Python script: $serializeErrorStream")
}
```

python

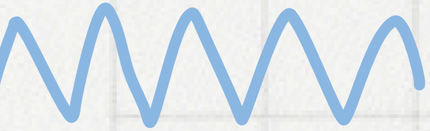
```python
capnpCalculatorSchema = capnp.load('calculator.capnp')

def serializeResponse(result):
    # Create the serialized response
    response = capnpCalculatorSchema.Response.new_message()
    response.result = result

    response_bytes = response.to_bytes()

    # Write the serialized response
    sys.stdout.buffer.write(response_bytes)
    sys.stdout.flush()
```

- Baut das Kotlin Projekt
- Lädt Dependencies
- Packetiert das Projekt (sorgt für Ausführbarkeit im Container)



Client
~Tizian

- Python Client
- Fast API -> Swagger UI
- Sendet Aufgaben im Cap'n Proto Format

```
def sendRequest(operation, num1, num2):
    request = capnpCalculatorSchema.Request.new_message()
    request.operation = operation
    request.num1 = num1
    request.num2 = num2
    requestBytes = request.to_bytes()
```

4. Fazit

~Tizian

Learnings

01

Umgang mit Capn Proto

02

Sackgassen Akzeptieren

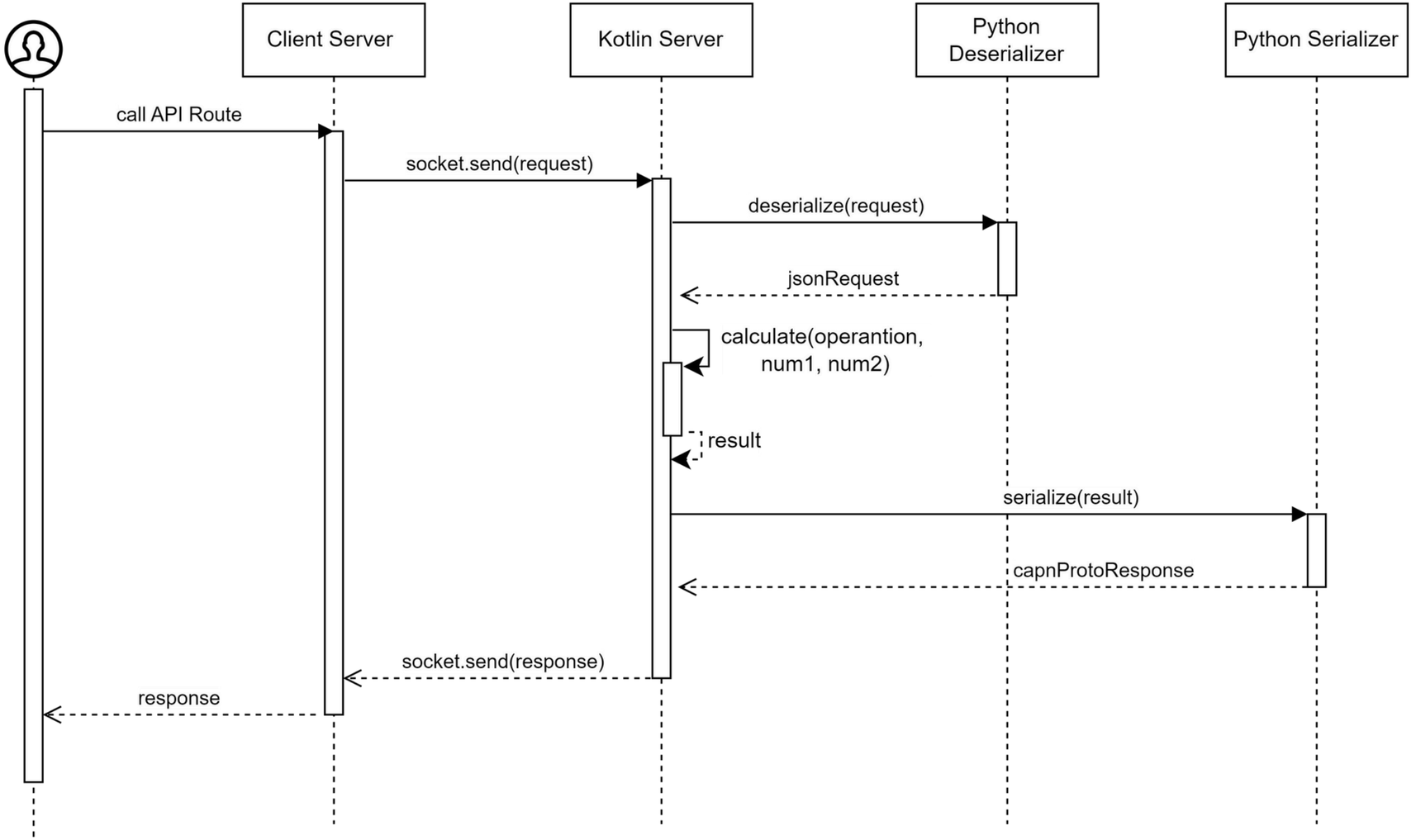
03

Tieferer Einblick in RPCs

04

Compilerbau


DAS WAR ES MIT UNSERER PRÄSENTATION
HABT IHR NOCH FRAGEN ?
makeameme.org


Client Server
Kotlin Server
Python Deserializer
Python Serializer
call API Route
socket.send(request)
deserialize(request)
jsonRequest
calculate(operantion, num1, num2)
result
serialize(result)
capnProtoResponse
socket.send(response)
response