CAP'N PROTO

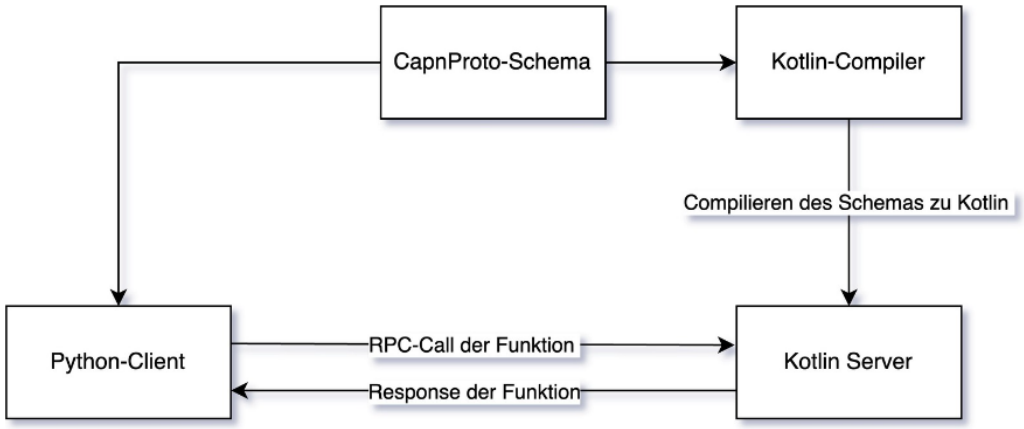
cerealization protocol

Projekt:	Verteilte Systeme / Compiler - Captain Kot
Dozenten:	Volker Birk
Gruppenmitglieder:	Michael Schick, Marcel Hasselberg, Tizian Grossmann
Studienjahrgang:	INF2022

1. Einleitung

Captain Kot besteht aus einem Python-Client und einem Kotlin-Server, die über das Cap'n Proto-Serialisierungsprotokoll miteinander kommunizieren. Dieses Protokoll sorgt für eine effiziente und schnelle Datenübertragung zwischen den beiden Komponenten. Ein wesentlicher Bestandteil von Captain Kot ist der Compiler, der Cap'n Proto-Schemas in Kotlin-Interfaces übersetzt. Dadurch können Entwickler einfach typsichere Datenmodelle in Kotlin verwenden, ohne die Strukturen manuell definieren zu müssen.

2. Architektur



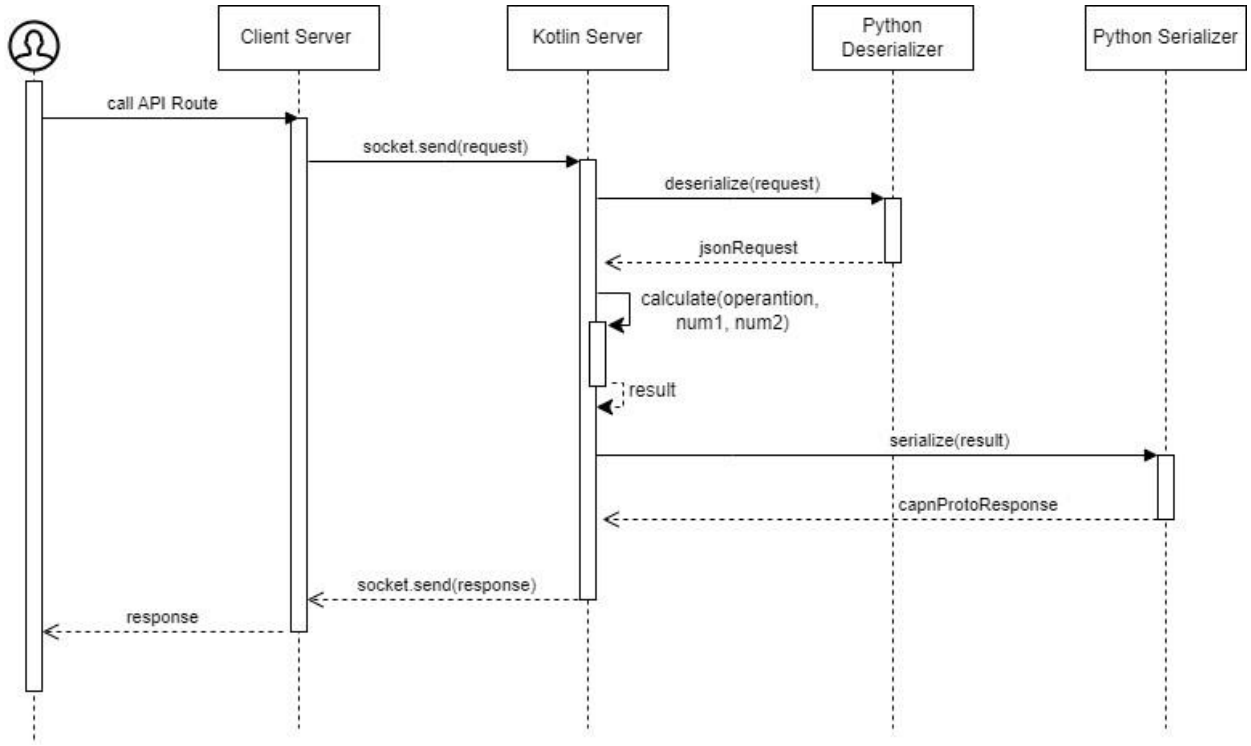
3. Umsetzung

Server: Zunächst muss das Cap'n Proto-Schema kompiliert werden. Der Compiler erzeugt dabei aus dem Schema Kotlin-Interfaces, die anschließend in den entsprechenden Kotlin-Quellcode eingefügt werden. Nachdem der generierte Code in das Projekt integriert ist, wird dieser mithilfe von Maven in ein ausführbares JAR-Paket umgewandelt

Client: Python Client mit API für die Taschenrechner Funktionen Addieren, Subtrahieren, Multiplizieren, Divideren. Diese sind in Swagger dokumentiert. Der Client schickt Anfragen im Cap'n Proto-Schema Fromat an den Server

Docker: das Ganze Projekt wurde Containerisiert und wird in der Docker Compose Umgebung ausgeführt.

4. Sequenzdiagramm



5. Reflexion

CaptainKot war ein erfolgreiches Projekt, das wertvolle Einblicke in die Nutzung von Cap'n Proto und die Kommunikation zwischen Python-Client und Kotlin-Server mithilfe eines RPCs ermöglichte. Eine zukünftige Weiterentwicklung könnte darin bestehen, die Serialisierung und Deserialisierung direkt im Kotlin-Server zu integrieren, um die Effizienz und Performance weiter zu verbessern.