Ira A. Fulton Schools of Engineering
Arizona State University
Engineering | School of Computing, Informatics, and Decision Systems Engineering

CSE 531: Distributed and Multiprocessor Operating Systems

gRPC Written Report

Problem Statement

Build a banking system where multiple branch servers and clients communicate with each other through gRPC and perform simple banking operations like query, deposit and withdraw. The system should also support inter-branch communications to propagate deposits and withdrawals to maintain consistency. For simplicity, it is assumed all customers share the bank account and a customer only interacts with a particular branch.

Goal

The goal of the problem statement is to build a distributed banking system using gRPC which provides basic banking functionality like balance query, deposit, withdrawal while maintaining consistency across all branches. It involves understanding and utilizing gRPC protocols to communicate between multiple branch servers(server-server) and customer clients(client-server) to perform banking operations.

Setup

Tech Stack:

- **Python** 3.9
- **gRPC** 1.58.0
- **Protobuf** 4.24.3
- **IDE** : IDEA IntelliJ 2023.2 Community Edition
- **Operating System** : macOS Ventura 13.5.2
- **Github** : Free,Pro, and Team

Implementation Processes

1. Define the protocol buffer messages representing the data structures and communication protocols between servers and clients
2. Use the Protobuf file to generate the gRPC server stub code in python
3. Implement the server and gRPC services extending generated service stub and create methods for Query, Deposit and Withdraw to handle client requests

4. Implement client code to connect to respective servers and invoke gRPC banking operations
5. Implement run_branch class to read the input file, separate branch events and start multiple servers depending on their branch ids and initialize their balances
6. Implement run_customer to read the input file, separate customer events and perform banking operations and store their results in an output file
7. Execute run_branch and run_customer on the inputs given in Project Description
8. Test the code against input file given as test case in canvas modules
9. Verify the banking operations are functionally correct and validate the test cases
10. Push final code to github with README file explaining the running of the project

Results

1. **Understanding of gRPC concepts**
    - **Result** : Thorough understanding of the fundamentals of gRPC, such as protocol buffers, service definitions, and remote procedure calls
    - **Justification**: Through the course of the project, students have understood the fundamentals of gRPC which is crucial to understand modern communication protocols used in various software applications
2. **Protobuf Message Definition**
    - **Result**: Students can define Protobuf messages to structure data for communication between client and server
    - **Justification**: Creating Protobuf messages enhances understanding of data serialization and deserialization, a fundamental aspect of efficient communication
3. **Service and Method Definitions**
    - **Result**: Define gRPC services and methods, including their request and response types
    - **Justification**: Defining services and methods helps understand how APIs are structured and called in a distributed system
4. **Server and Client Implementation**
    - **Result**: Implementation of gRPC servers and clients for handling banking operations
    - **Justification**: Implementing servers and clients gives us hands-on experience in building networked applications using gRPC
5. **Functionality**
    - **Result**: The banking system supports essential functionalities like querying account balances, depositing funds, and withdrawing funds while keeping the balances consistent across all branches
    - **Justification**: Fulfilling these fundamental banking operations ensures that the system serves its purpose of enabling users to manage their accounts effectively