RPC-based File Transfer System

1 Design of the RPC Service

The Remote Procedure Call (RPC) service is designed using gRPC, with service definitions and message structures specified in a Protocol Buffers (‘.proto‘) file. The design includes two main RPC methods:

- **ListFiles**: Returns a list of files available on the server.
- **DownloadFile**: Streams the content of a requested file from the server to the client.

Service Design (Figure 1):

```
+--------------------------------------+
|            FileTransfer Service      |
|--------------------------------------|
|  + ListFiles(Empty) -> FileList      |
|  + DownloadFile(FileRequest)         |
|    -> stream FileChunk               |
+--------------------------------------+
```

1.1 Protocol Buffers Definition

The Protocol Buffers definition is as follows:

```
syntax = "proto3";

service FileTransfer {
  rpc ListFiles(Empty) returns (FileList);
  rpc DownloadFile(FileRequest) returns (stream FileChunk);
}

message Empty {}

message FileList {
  repeated string filenames = 1;
}

message FileRequest {
  string filename = 1;
}

message FileChunk {
  bytes content = 1;
}
```

Listing 1: file_transfer.proto

2 System Organization

The system consists of two primary components: the server and the client. The server hosts the files and provides the RPC service, while the client communicates with the server to list and download files.

System Organization (Figure 2):

```
+-------------+      gRPC      +-------------+
|             | <============> |             |
|    Client   |                |    Server   |
|             |                |             |
+-------------+                +-------------+
| List Files  |                | Provide List|
| Download    |                | Stream File |
+-------------+                +-------------+
```

3 Implementation of File Transfer

The file transfer functionality is implemented using gRPC streaming, where the server streams chunks of the file to the client. Below are the key code snippets for the implementation:

3.1 Server Implementation

```python
class FileTransferService(file_transfer_pb2_grpc.FileTransferServicer):
    def ListFiles(self, request, context):
        files = os.listdir(SERVER_DIRECTORY)
        return file_transfer_pb2.FileList(filenames=files)

    def DownloadFile(self, request, context):
        file_path = os.path.join(SERVER_DIRECTORY, request.filename)
        if not os.path.exists(file_path):
            context.abort(grpc.StatusCode.NOT_FOUND, "File not found")

        with open(file_path, 'rb') as f:
            while chunk := f.read(BUFFER_SIZE):
                yield file_transfer_pb2.FileChunk(content=chunk)
```

Listing 2: Server Implementation

3.2 Client Implementation

```python
def download_file(stub, filename):
    file_path = os.path.join(CLIENT_DIRECTORY, filename)
    response = stub.DownloadFile(file_transfer_pb2.FileRequest(filename=filename))
    os.makedirs(CLIENT_DIRECTORY, exist_ok=True)
    with open(file_path, 'wb') as f:
        for chunk in response:
            f.write(chunk.content)
```

Listing 3: Client Implementation

4 Roles and Responsibilities

- **Server:**
  - Hosts the files in a directory.
  - Implements the gRPC service methods `ListFiles` and `DownloadFile`.
  - Streams file chunks to the client.
- **Client:**
  - Requests the list of available files.
  - Selects a file for download.
  - Receives the streamed file chunks and reconstructs the file locally.