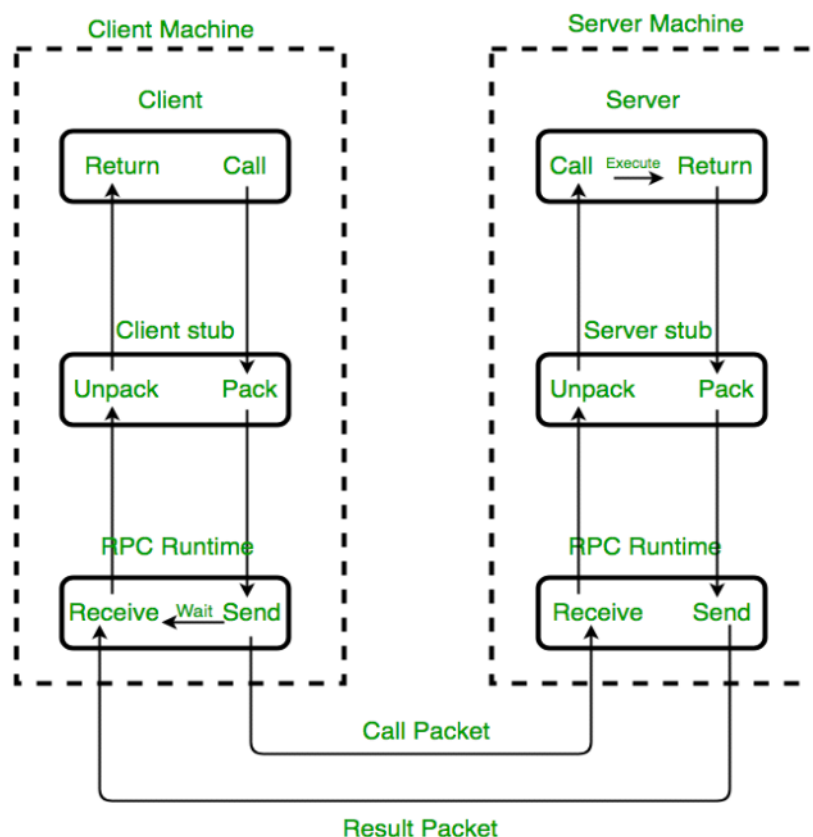


# 第80期-使用gRPC实现远程过程调用

2020年1月11日 14:19

## RPC 远程过程调用

两台机器，A 机器上的程序要调用 B 机器上某程序提供的函数或方法，由于不在一个内存空间，不能直接调用，需要通过网络来表达调用的语义和传达调用的数据。



谁能用通俗的语言解释一下什么是RPC 框架？

## gRPC 技术简介

gRPC 是一种与语言无关的高性能远程过程调用 (RPC) 框架。

## 技术优点

- 现代高性能轻量级 RPC 框架。
- 协定优先 API 开发，默认使用协议缓冲区，允许与语言无关的实现。
- 可用于多种语言的工具，以生成强类型服务器和客户端。
- 支持客户端、服务器和双向流式处理调用。
- 使用 Protobuf 二进制序列化减少对网络的使用。

## 这些优点使 gRPC 适用于

- 效率至关重要的轻量级微服务。
- 需要处理流式请求或响应的点对点实时服务。

## 创建 gRPC 服务端

### 先决条件

- Visual Studio 2019 + 与 ASP.NET 和 Web 开发 工作负载
- .NET Core 3.0 SDK +

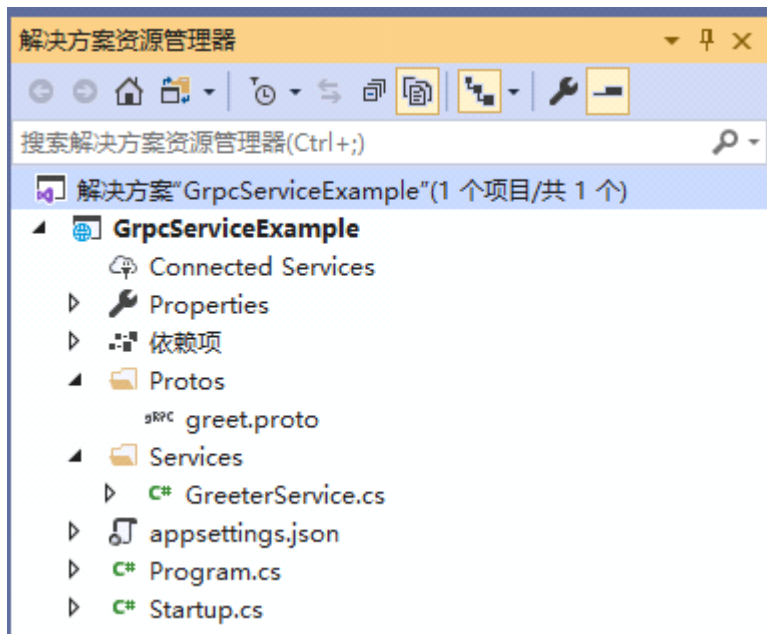
### 创建 gRPC 服务



### 运行服务

按 Ctrl+F5 以在不使用调试程序的情况下运行。

### 项目文件说明



## 创建 gRPC 客户端

### 客户端项目需要依赖的包

```
Install-Package Grpc.Net.Client  
Install-Package Google.Protobuf  
Install-Package Grpc.Tools
```

### 添加 greet.proto 文件

```
<ItemGroup>  
  <Protobuf Include="Protos\greet.proto" GrpcServices="Client" />  
</ItemGroup>
```

### 客户端代码编写

```
static async Task Main(string[] args)  
{  
    // The port number(5001) must match the port of the gRPC server.  
    using var channel = GrpcChannel.ForAddress("https://localhost:5001");  
    var client = new Greeter.GreeterClient(channel);  
    var reply = await client.SayHelloAsync(new HelloRequest { Name =  
"GreeterClient" });  
    Console.WriteLine("Greeting: " + reply.Message);  
    Console.WriteLine("Press any key to exit...");  
    Console.ReadKey();  
}
```