# 图书下载链接

#### 全国图书馆

[*http://www.ucdrs.superlib.net/*](http://www.ucdrs.superlib.net/)

#### 龙岩图书馆

[*http://www.ly.superlib.net/*](http://www.ly.superlib.net/)

#### 深圳文献港

[*http://search.szdnet.org.cn/*](http://search.szdnet.org.cn/)

# RPC简介及框架选择

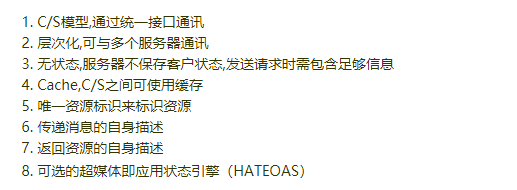
参考链接 https://blog.csdn.net/en\_joker/article/details/105556828

## 传统Restful Api

一篇不错的理解<https://www.zhihu.com/question/27785028/answer/48096396>

Restful Api 是一种符合REST架构设计的API 是一种设计风格 符合REST设计标准的API就是Restful Api

REST的主要原则有以下几点



1. C/S模型 通过统一接口通讯
2. 无状态 服务器不保存客户状态 发送请求时候 需要包含足够信息？

REST: Representational State Transfer (资源的)表现层状态转化

REST的几个核心概念:

1. Resources 资源
2. Representation 表现层 资源具体呈现的方式 比如 一段文本 可以用 txt格式展现 也可以用html格式、xml、json格式展现

3）状态转化 可以理解为 资源 由一种展现形式 转换到 另外一种展现形式

客户端如需操作资源 进行状态转化 可以通过常用的四个动词 GET、POST、PUT、DELETE

## IPC

IPC(Inter-Process-Communication) 是一种UNIX的通信机制

共分为两类 LPC和RPC

### LPC

Local-Procedure-Call 本地过程调用 详情请参考操作系统 进程间通信方式以及原理

### RPC

Remote-Procedure-Call 远程过程调用

核心 是 在本地调用远程的方法

RPC的核心并不在于使用什么协议。RPC的目的是让你在本地调用远程的方法，而对你来说这个调用是透明的，你并不知道这个调用的方法是部署哪里。通过RPC能解耦服务，这才是使用RPC的真正目的。RPC的原理主要用到了动态代理模式，至于http协议，只是传输协议而已。简单的实现可以参考spring remoting，复杂的实现可以参考dubbo。

简单的说，

* RPC就是从一台机器（客户端）上通过参数传递的方式调用另一台机器（服务器）上的一个函数或方法（可以统称为服务）并得到返回的结果。
* RPC 会隐藏底层的通讯细节（不需要直接处理Socket通讯或Http通讯） RPC 是一个请求响应模型。
* 客户端发起请求，服务器返回响应（类似于Http的工作方式） RPC 在使用形式上像调用本地函数（或方法）一样去调用远程的函数（或方法）。

#### 常用的RPC框架

Netty :不局限于RPC用途 更多的是作为一种网络协议的实现框架 比如HTTP 由于RPC需要高效的网络通信 就可能选择以Netty为基础

Brpc 是一个基于protobuf接口的RPC框架

Dubbo 是Alibaba开发的一个RPC框架 远程接口给予java interface 依托于spring框架

gRPC

Thrift 是Appache的一个项目 前身是Facebook开发的一个RPC框架 采用thrift作为 接口描述语言

Jsonrpc

##### JSON-RPC

[python web接口实现（restful方式、jsonrpc方式）](https://www.jianshu.com/p/545acae57e27)

* 区块链项目中用的较多？资料不是很多
* JSON-RPC是一种序列化协议。JSON 是 JS 对象的字符串表示法，它使用文本表示一个 JS 对象的信息，本质是一个字符串。
* 非常简单，方便，速度慢
* 相关Python 包(直接集成到flask和django)
* Flask-JSONRPC,django-json-rpc；jsonrpcserver,jsonrpcclient

##### Thrift

[Python RPC 之 Thrift](https://www.jianshu.com/p/82a6bdaabcd3)

* Facebook开源的跨语言RPC框架。

##### gRPC

总的来说，Python的 rpc框架选择较少，thrift性能最好，grpc性能比thrift稍差，原因是多了http2，而thrift直接基于tcp，但grpc序列化方案更通用(protobuf)优秀，文档较好；  
jsonrpc 本身基于http/1进行通信，速度最慢，相对于之前速度无提升，只是接口和数据格式更为统一；

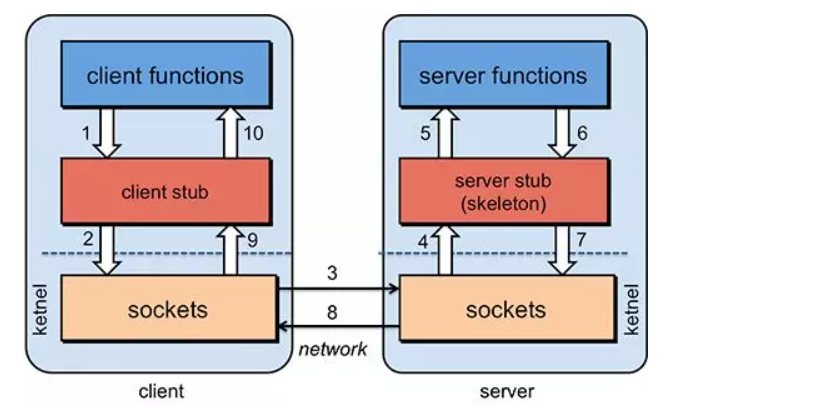
个人认为 之所以项目选择gRPC作为RPC框架 有以下几点原因:

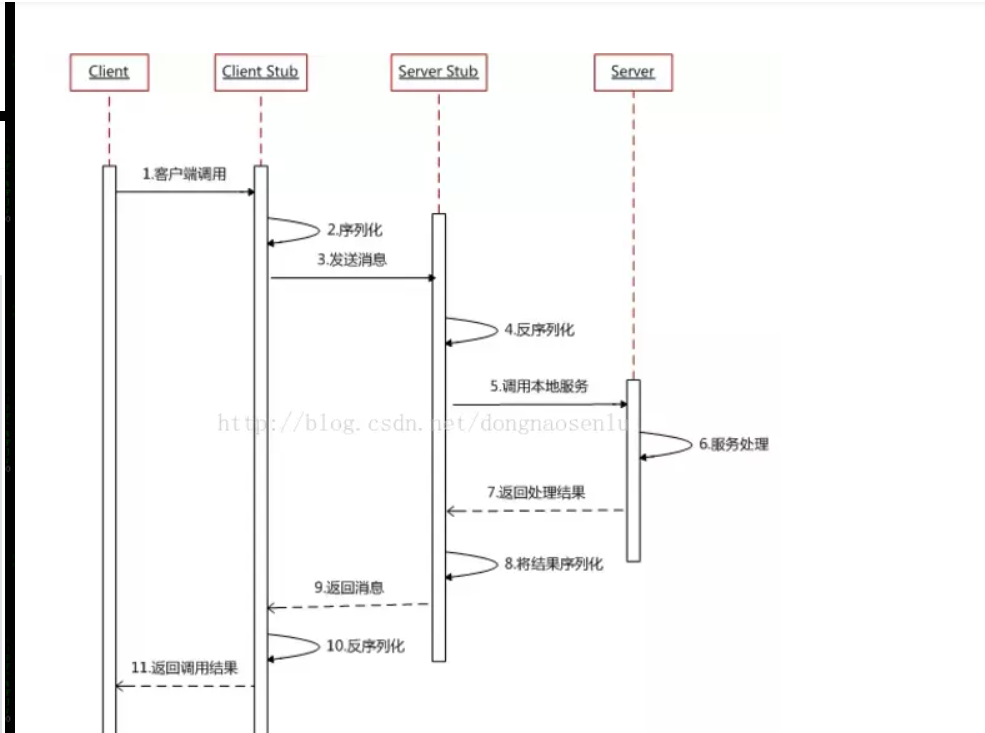
1. 采用protobuf序列化方案，该方案较为优秀（平台下发参数时候 以及 自动化返回结果都需要进行序列化和反序列化惭怍）
2. 给予http2进行通信 http2 天然支持物联网、手机、浏览器 并且安全性较高

但是http2也有相应的缺点: 1 rpc的元数据传输不够高效 2 http2里一次gRPC调用需要解码两次 一次是Headers frame 一次是DataFrame

Grpc给予http2 进行通讯 那个他的性能肯定不是最顶尖的，但是对于rpc来说 中庸的qps可以接受，通用和兼容才是最重要的事情。

#### 通讯过程





##### 相关名词

Ketnel : 操作系统内核 操作系统内核是指大多数操作系统的核心部分。它由操作系统中用于管理存储器、文件、外设和系统资源的那些部分组成。操作系统内核通常运行进程，并提供进程间的通信。下面列出了它的一些核心功能：事件的调度和同步。进程间的通信(消息传递)。存储器管理。进程管理。输入输出例程的管理。

序列化协议：

常见的xml、json、和比较新的protocol buffers 、Thift

传输协议

比如 TCP、UDP 以及建立在TCP和UDP基础上的应用层协议 http

Tcp/UDP和http的关系 参考链接、

<https://www.cnblogs.com/baizhanshi/p/8482612.html>