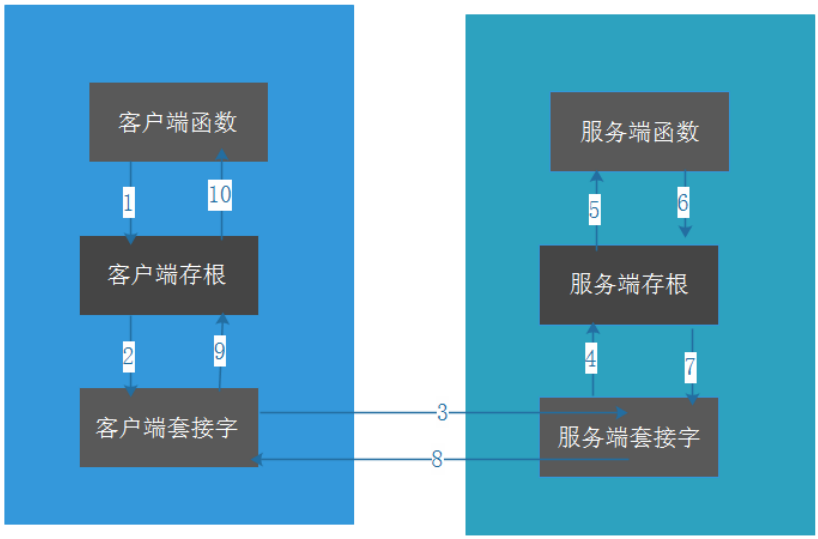
07_RPC服务的发展历程

- 1，RPC的特点：
  - 随着移动互联网的高速发展，面对不断增长的用户量及海量的请求，后端基本是规模庞大的分布式集群
  - 其中，RPC（远程过程调用），便是其中的核心技术
  - RPC是一种通过网络，向远程计算机请求服务，而不需要了解其底层网络技术的协议。
  - RPC协议，以传输协议为基础（TCP,UDP），为两个不同的应用程序间传递数据。
    - TCP，建立可靠连接，三次握手，四次挥手
    - UDP,没有建立连接，不可靠的传输协议
  - 在网络参考模型中，RPC位于传输层和应用层之间
  - RPC采用客户端/服务端工作模式。
- 2，RPC与RESTful的差异
  - 调用RESTful服务时，我们需要采用HttpClient的方式发送请求，同时需要对返回的数据去解析状态，返回值等等
  - 采用RPC的方式调用服务，则如同调用本地服务一样，不必关心网络通信的细节
- 3，RPC的工作原理
  - Socket套接字
    - 网络上的两个程序，通过一个双向的通信连接，实现数据的交换，这个连接的一端就称为套接字
    - Socket包含两部分（IP地址+端口号）
    - Java为我们提供的Scoket分为客户端和服务端，分别是Socket和ServerScoket
  - 本地调用是如何实现的？
    - 首先，程序被编译成具体CPU提供的机器指令
    - 然后，在调用本地程序时，先将下一条指令的地址压入堆栈，并将控制权转移到当前调用程序的地址
    - 当被调用的程序完成时，它会发出一个返回指令，并从栈顶弹出之前保存的地址，并将控制权转移回来。
    - A->B->C
  - 远程调用会遇到的问题
    - 以上的本地调用机制在RPC是不可行的，因为编译器无法通过编译的方法实现远程过程调用机制
    - 因此，RPC远程调用是构建在语言级别上的，必须使用Socket来完成。
    - 所以RPC的实现=本地方法调用+Scoket网络通信技术
  - 具体如何实现？
    - 关键是创建“客户端存根”，存根就像是代理模式中的代理，在生成代理代码后，代理的代码就可以跟远程服务端通信了
    - 具体通信的图如下：
    -



- 4，RPC主流的框架
  - 阿里巴巴的HSF(好舒服)，Dubbo（开源）
  - Facebook的Thrift（开源）
  - Google的gRPC（开源）
  - Twitter的Finagle（开源）
  - 微博平台的Motan
  - mRPC
- 5，带来的好处
  - 我们编写分布式程序将变得简单，无需关注底层的通讯机制，套接字，端口，数据转换等细节问题
- 6，Java与RPC
  - Java RMI（Remote Method Invocation，远程方法调用），是java专有的RPC协议和框架。