RPC（**Remote Procedure Call**，远程过程调用）采用**客户端/服务端**模式，请求程序就是一个客户端，服务提供程序就是一个服务端。RPC框架帮我们把繁琐的通信过程和唤起调用服务的过程都包装好了，更便于开发人员使用。

首先，客户机调用进程发送一个有输入参数的调用信息给服务端的进程，并等待应答信息；然后，**服务端的进程保持睡眠状态直到调用信息到达**，当有一个调用信息到达时，服务端获得输入参数，计算结果或者进行一系列复杂操作，甚至调用更多的其他远程服务器；其次，服务端发送应答信息，等待下一个调用信息；最后，客户端调用进程接收到答复消息，获得输出的结果，再继续执行后续的操作。

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，这个连接的一端被称为**Socket**。Socket用于描述**IP地址**和**端口**，是一个通信连接的**句柄**，可以用来实现不同的计算机之间的通信，是网络编程接口的具体实现。

在本地过程调用的过程中，程序被编译器编译为具体的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，同时将控制权转移回来**。

实现透明的远程过程调用的重点是创建客户**存根**（client stub），存根（stub）就像代理（agent）模式里的**代理**，**在生成代理代码后，代理的代码就能与远程服务端通信了，通信的过程都由RPC框架实现，而调用者就像调用本地代码一样方便**。在客户端看来，存根函数就像普通的本地函数一样，但实际上包含了通过网络发送和接收消息的代码。