**一、优点**

在移动设备上（性能受限制的设备，比如要省电），广泛地使用跨进程通信对通信机制的性能有严格的要求，**Binder**相对出传统的Socket方式，更加高效。Binder数据拷贝只需要一次，而**管道、消息队列、Socket**都需要2次。共享内存方式一次内存拷贝都不需要，但实现方式又比较复杂。

传统的进程通信方式对于通信双方的身份并没有做出严格的验证，比如Socket通信ip地址是客户端手动填入，很容易进行伪造，而Binder机制从协议本身就支持对通信双方做身份校检，因而大大提升了安全性。

**二、四大元素**

Server

Client

ServiceManager：client通过它获得Server中Binder实体的引用

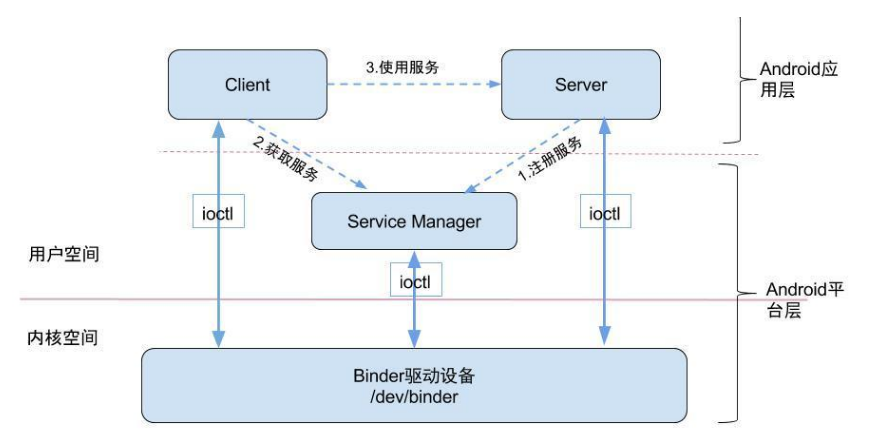
Binder驱动：工作于工作内核，负责Binder在进程间的传递，Binder通信的建立

**三、运行机制**

注册服务：Server进程要先注册Service到ServiceManager。该过程：Server是客户端，ServiceManager是服务端

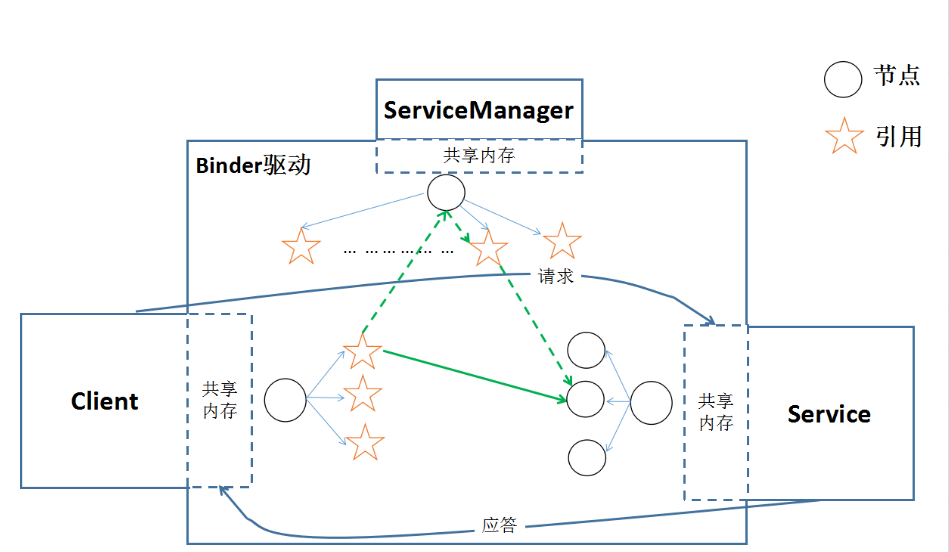
获取服务：Client进程使用某个Service前，须先向ServiceManager中获取相应的Service。该过程：Client是客户端，ServiceManager是服务端。

使用服务：Client根据得到的Service信息建立与Service所在的Server进程通信的通路，然后就可以直接与Service交互。该过程：client是客户端，server是服务端。



三者之间的通信都是通过Binder驱动

**四、client如何调用远程service的方法**



服务端跨进程的类都要继承Binder类。我们所持有的Binder引用(即服务端的类引用)并不是实际真实的远程Binder对象，我们的引用在Binder驱动里还要做一次映射。也就是说，设备驱动根据我们的引用对象找到对应的远程进程。客户端要调用远程对象函数时，只需把数据写入到Parcel，在调用所持有的Binder引用的transact()函数，transact函数执行过程中会把参数、标识符（标记远程对象及其函数）等数据放入到Client的共享内存，Binder驱动从Client的共享内存中读取数据，根据这些数据找到对应的远程进程的共享内存，把数据拷贝到远程进程的共享内存中，并通知远程进程执行onTransact()函数，这个函数也是属于Binder类。远程进程Binder对象执行完成后，将得到的写入自己的共享内存中，Binder驱动再将远程进程的共享内存数据拷贝到客户端的共享内存，并唤醒客户端线程。

参考资料：<http://www.jcodecraeer.com/a/anzhuokaifa/androidkaifa/2015/0319/2619.html>

<http://blog.csdn.net/huachao1001/article/details/51504469>

<http://www.jianshu.com/p/4920c7781afe?from=jiantop.com>

<http://blog.csdn.net/boyupeng/article/details/47011383>