MICROOH 麦可网

Android-从程序员到架构师之路

出品人：Sundy
讲师：高焕堂（台湾）

http://www.microoh.com

F01_a

观摩：Session模式与Proxy-Stub模式的搭配(a)

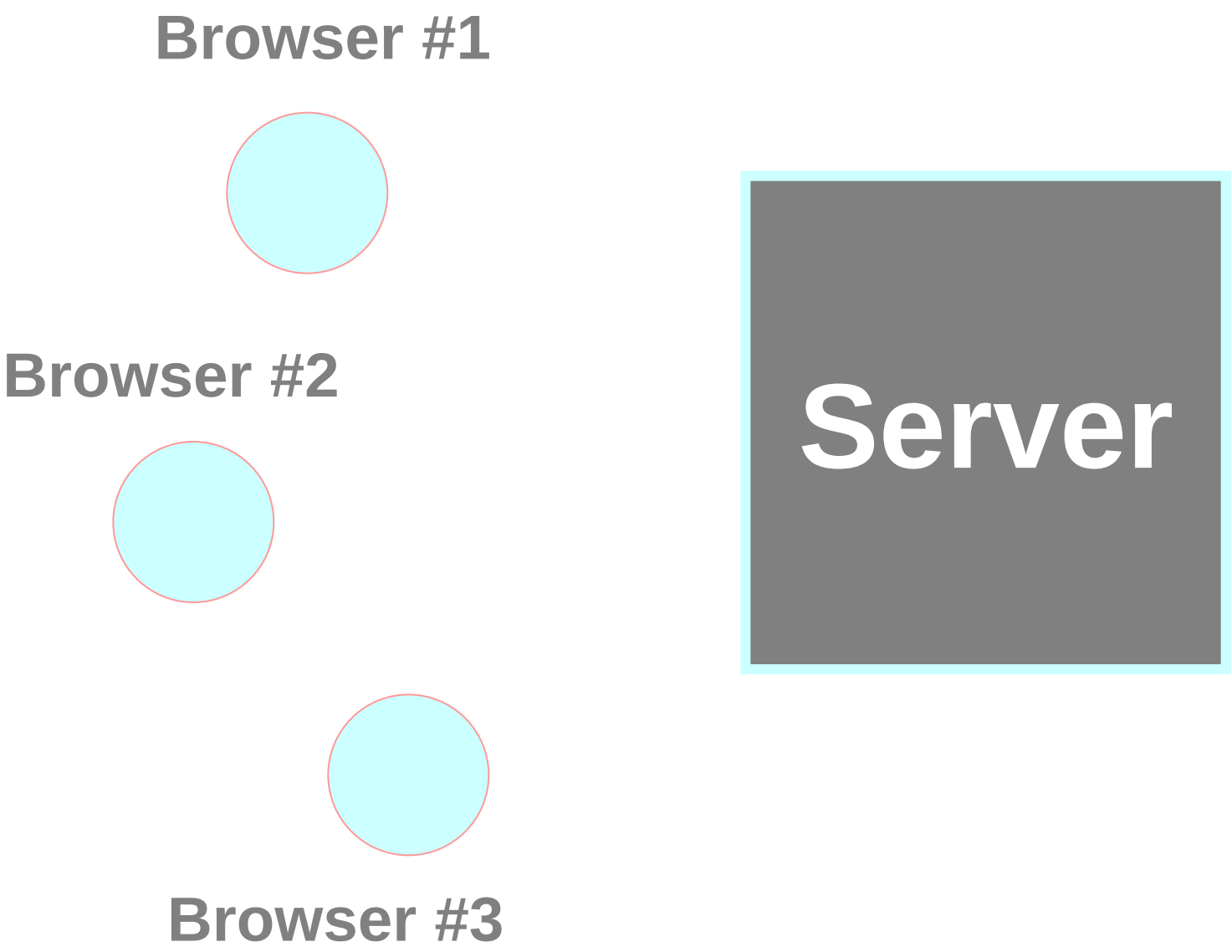
By 高煥堂

内容

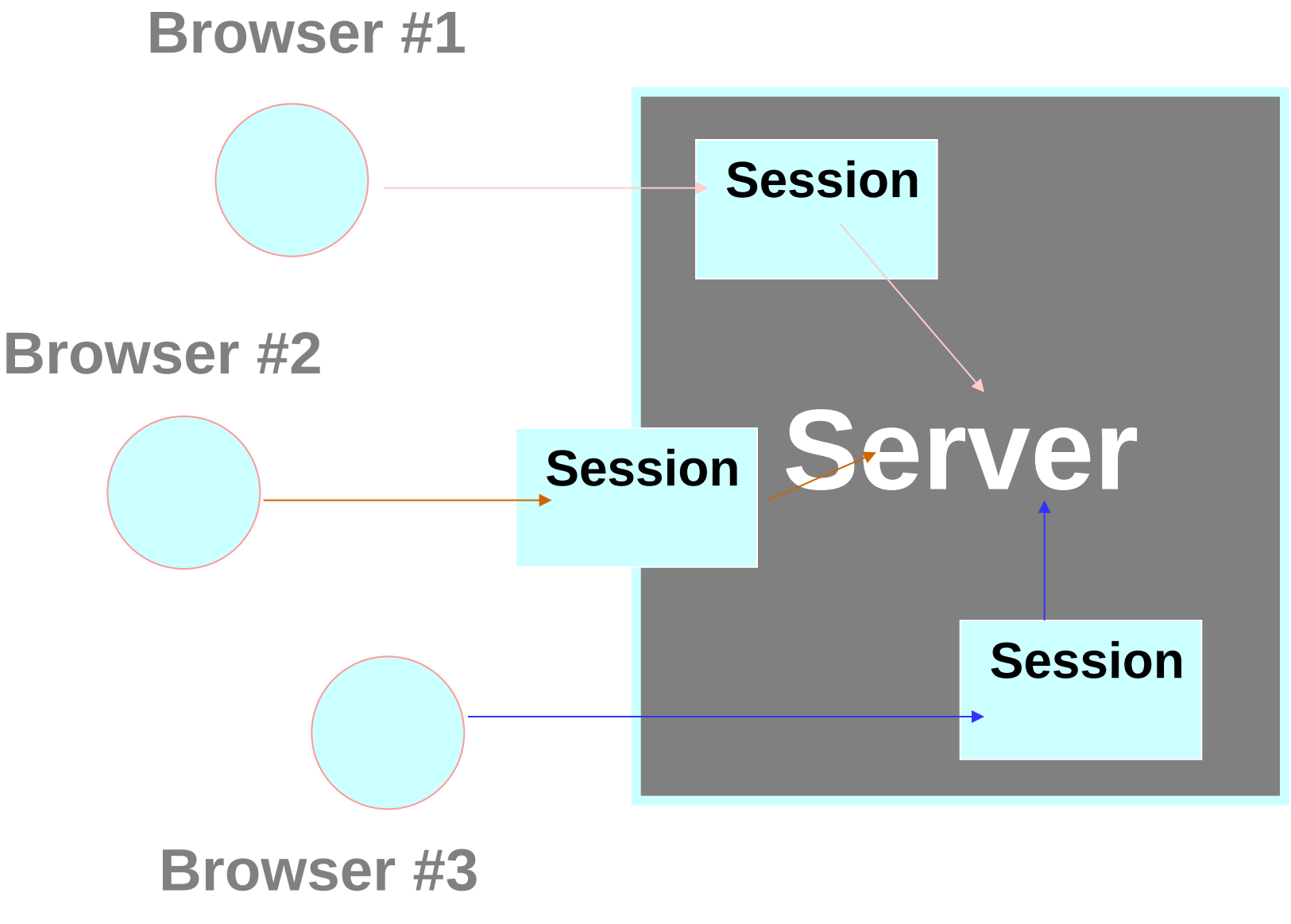
1. Session设计模式：以CameraService服务为例
2. Session设计模式：以VM的JNIEnv对象为例
3. Session设计模式：典型架构
4. 复习：Proxy-Stub模式
5. Proxy-Stub设计模式： 以CameraService为例
6. SurfaceFlinger服务的Session模式
7. SurfaceFlinger服务的Proxy-Stub模式

1、Session设计模式：以CameraService服务为例

前言


Browser #1
Browser #2
Browser #3
Server

每一个Connection
都有一个私有的Session对象；


Browser #1
Session
Browser #2
Session
Server
Session
Browser #3

以CameraService为例

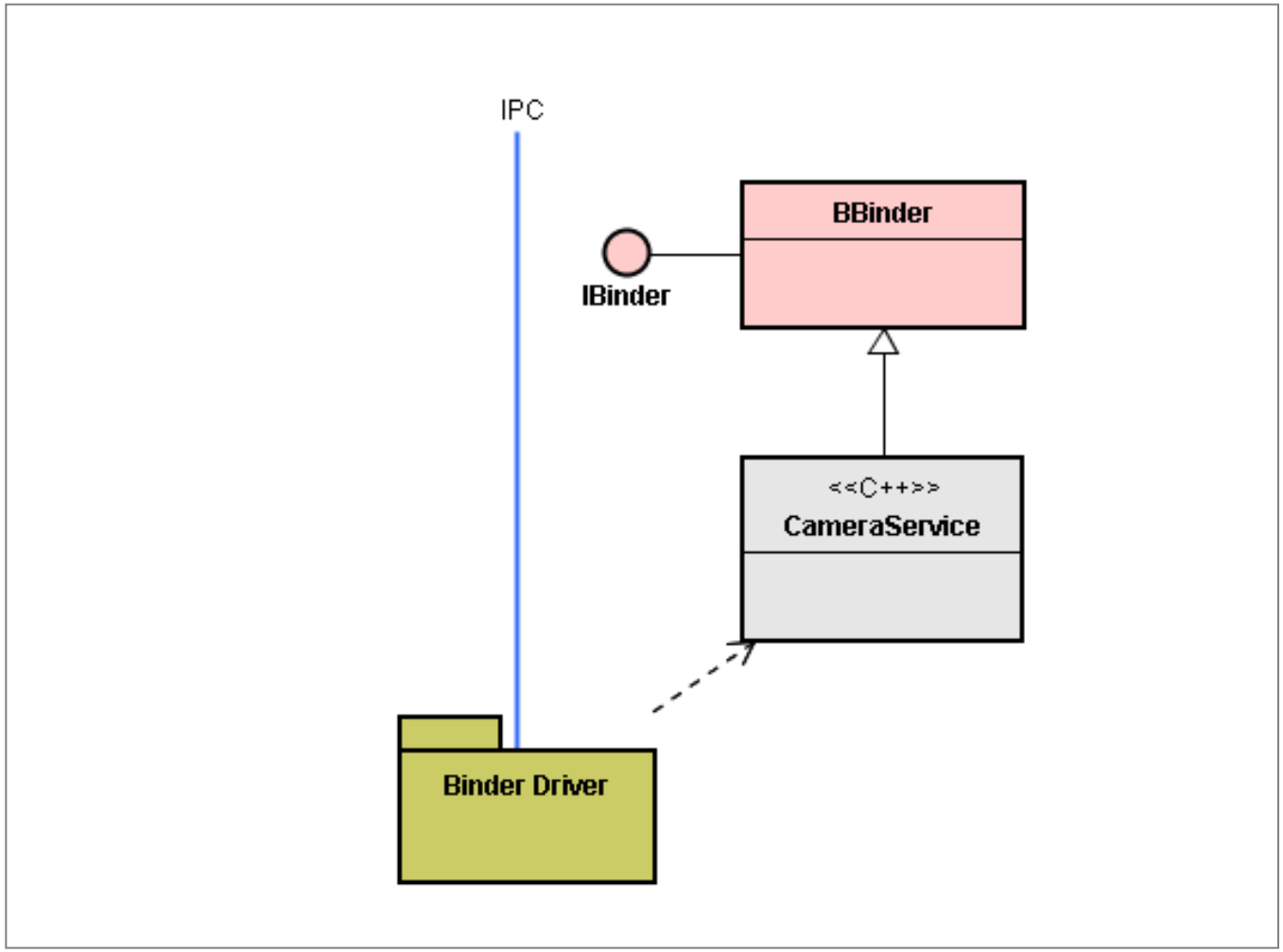
- 在Android的C++层里，有个CameraService系统服务。
- 在MediaServer进程初始化照相机服务。
- 此进程的main()函数代码：

```
int main(int argc, char** argv) {
    sp<ProcessState> proc(ProcessState::self());
    sp<IServiceManager> sm = defaultServiceManager();
        // .........
    CameraService::instantiate();
        // .........
    IPCThreadState::self()->joinThreadPool();
}
```

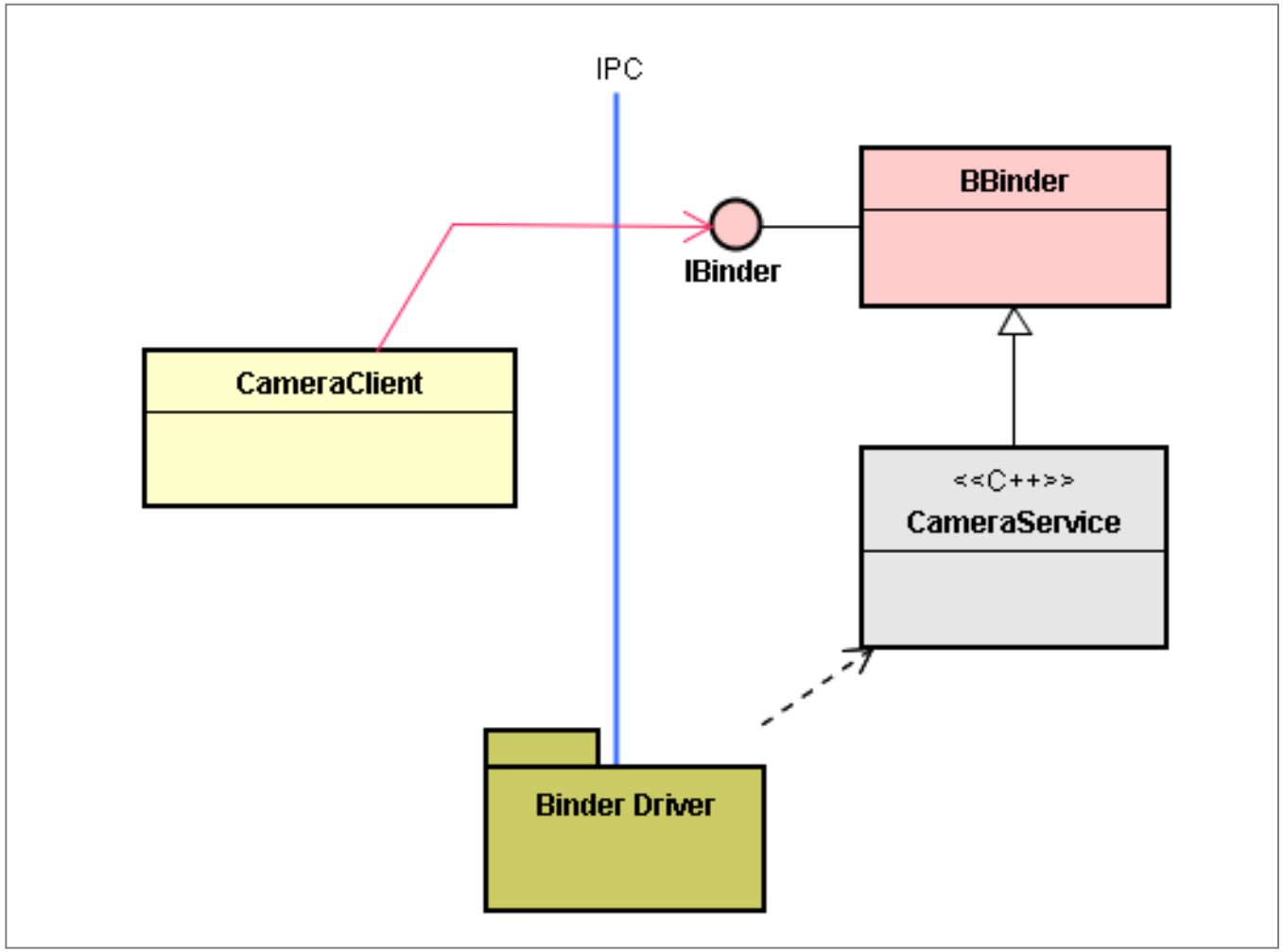
- 这个 CameraService::instantiate()的代码是：

```
void CameraService::instantiate() {
    defaultServiceManager()->addService(
            String16("media.camera"), new CameraService());
}
```

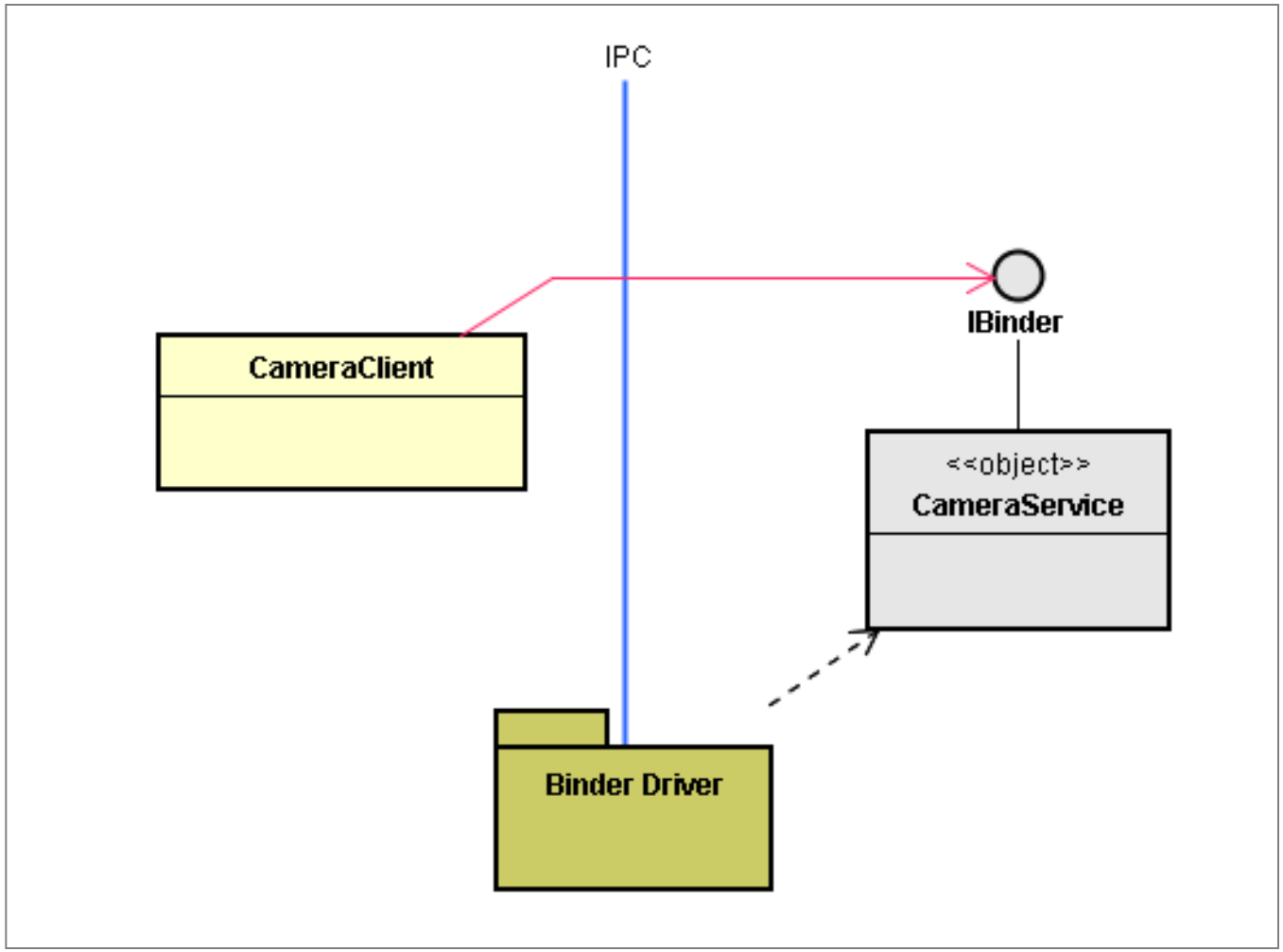
- 首先创建CameraService对象,然后委托SM(ServiceManager)登录到BD(Binder Driver)里。


IPC
BBinder
IBinder
<<C++>>
CameraService
Binder Driver

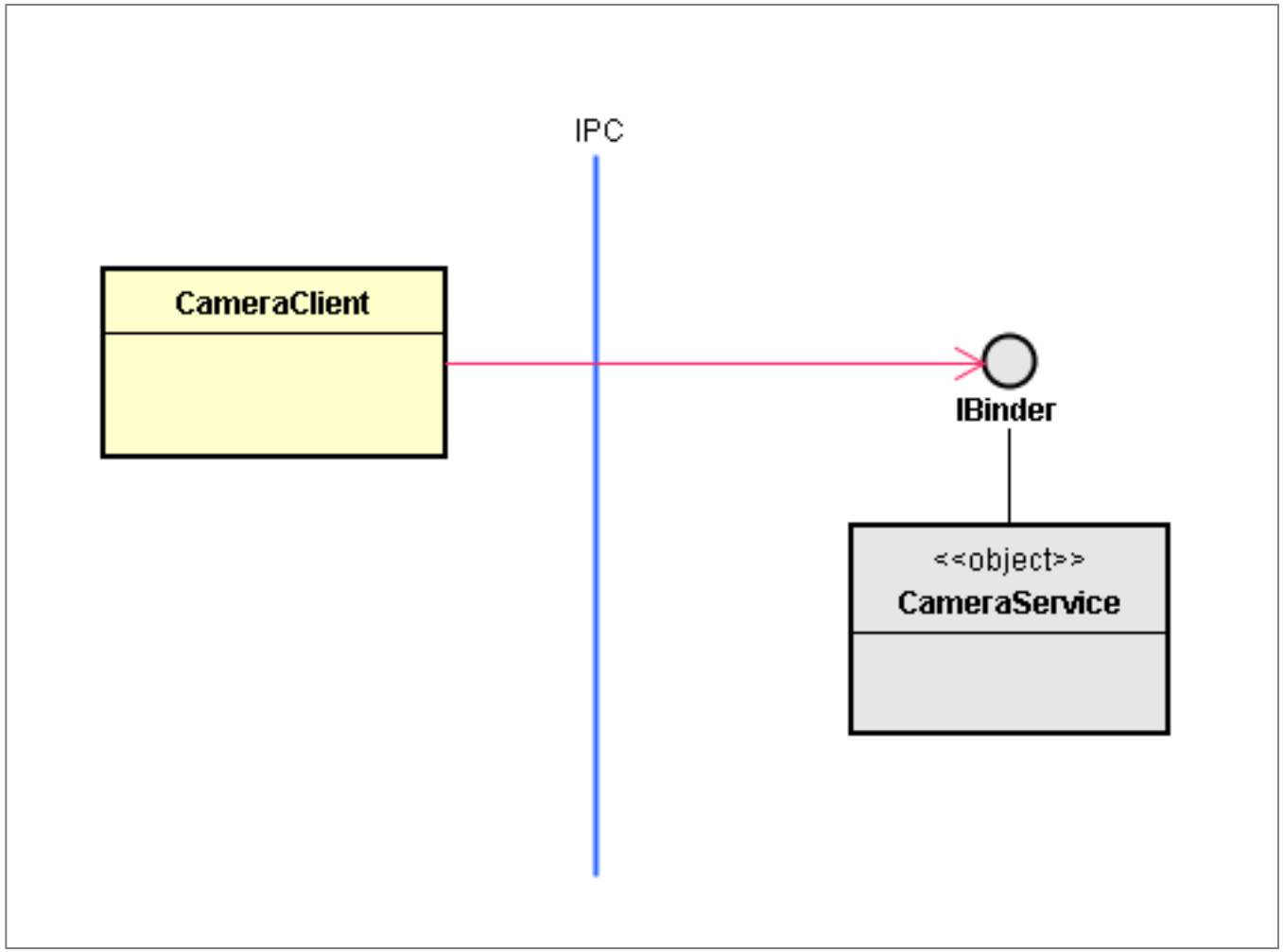
- 当一个Camera应用程序启动时，会建立一个CameraClient来与CameraService衔接。


IPC
BBinder
IBinder
CameraClient
<<C++>>
CameraService
Binder Driver

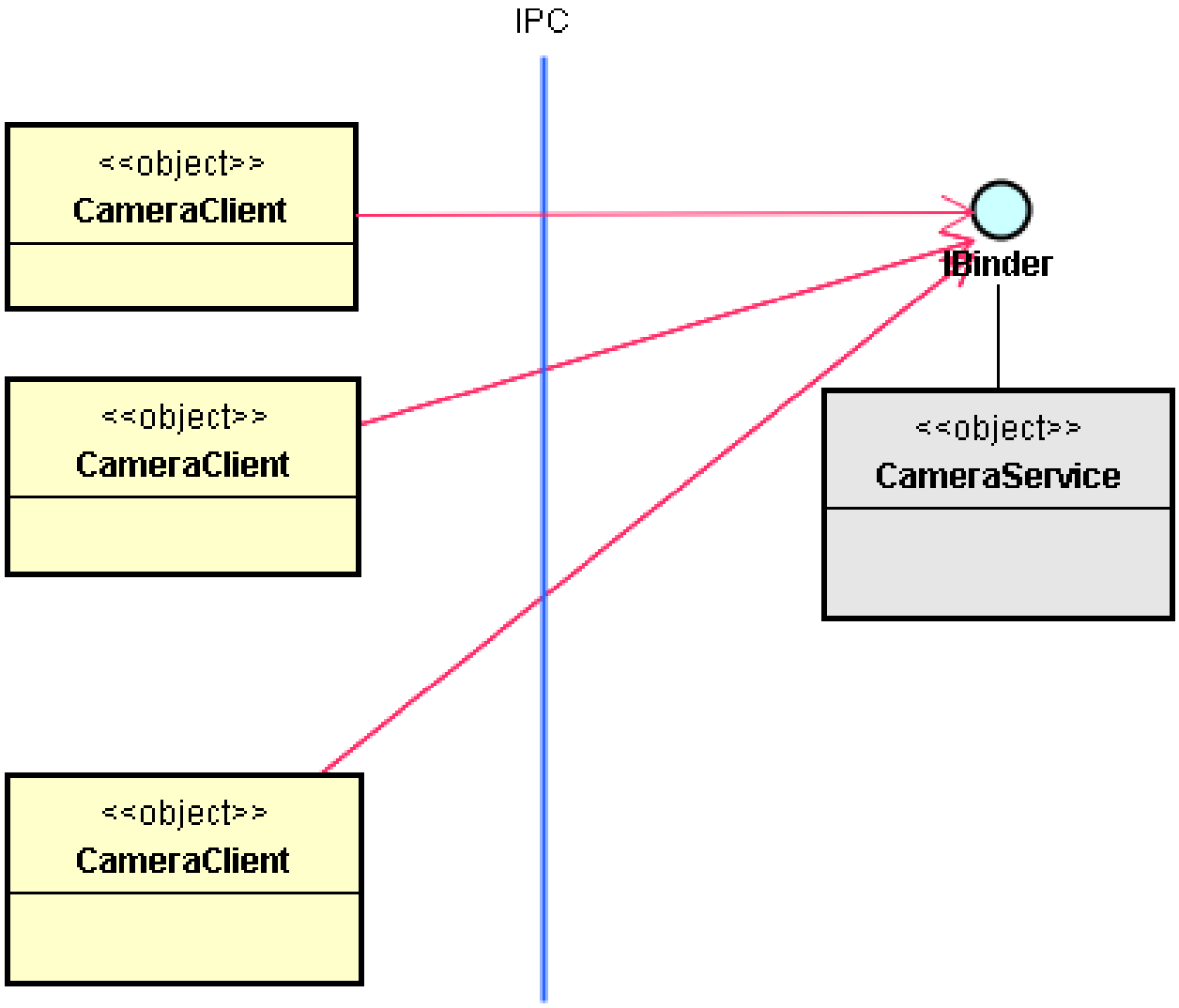
相当于：

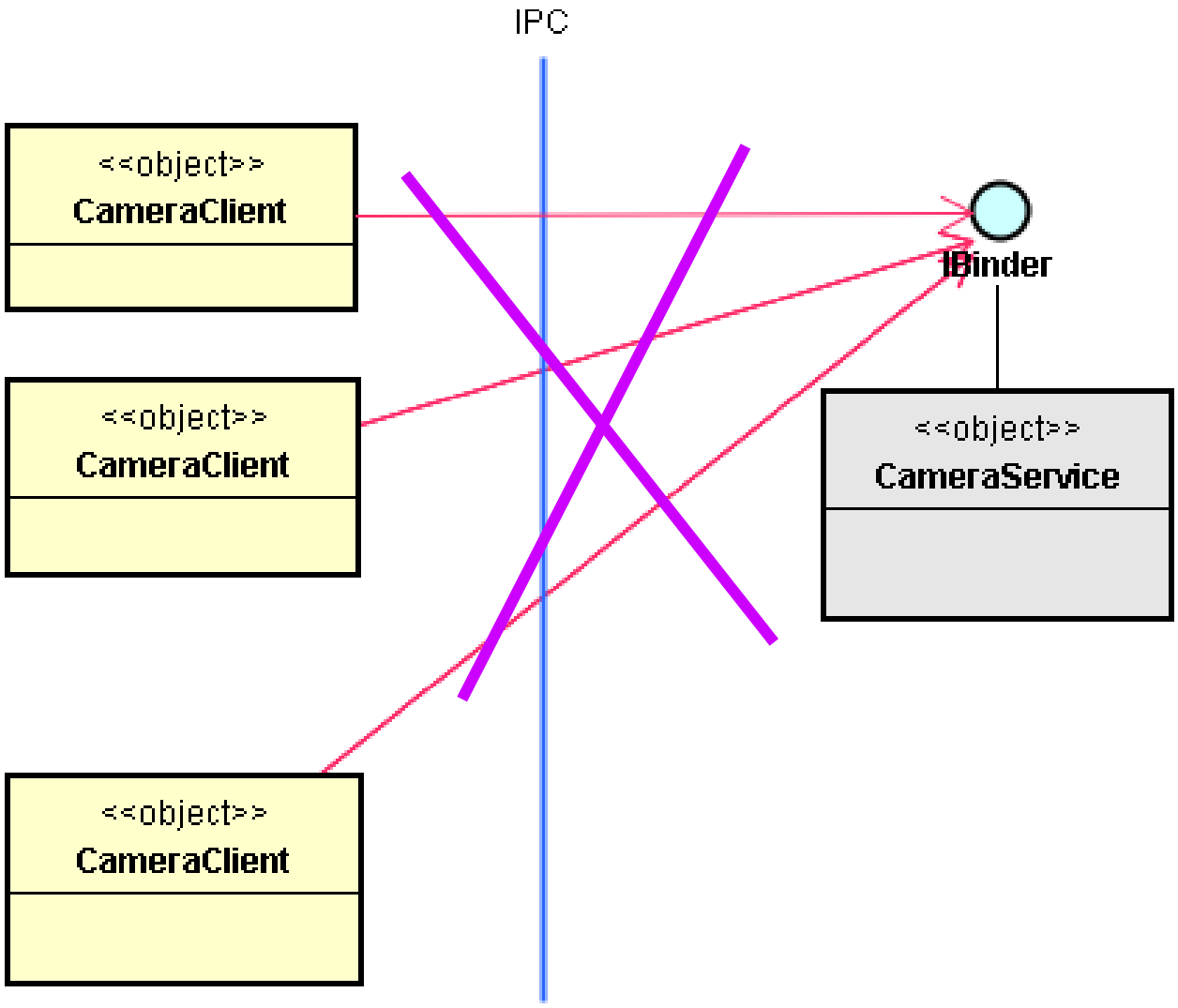

IPC
IBinder
CameraClient
<<object>>
CameraService
Binder Driver

相当于：



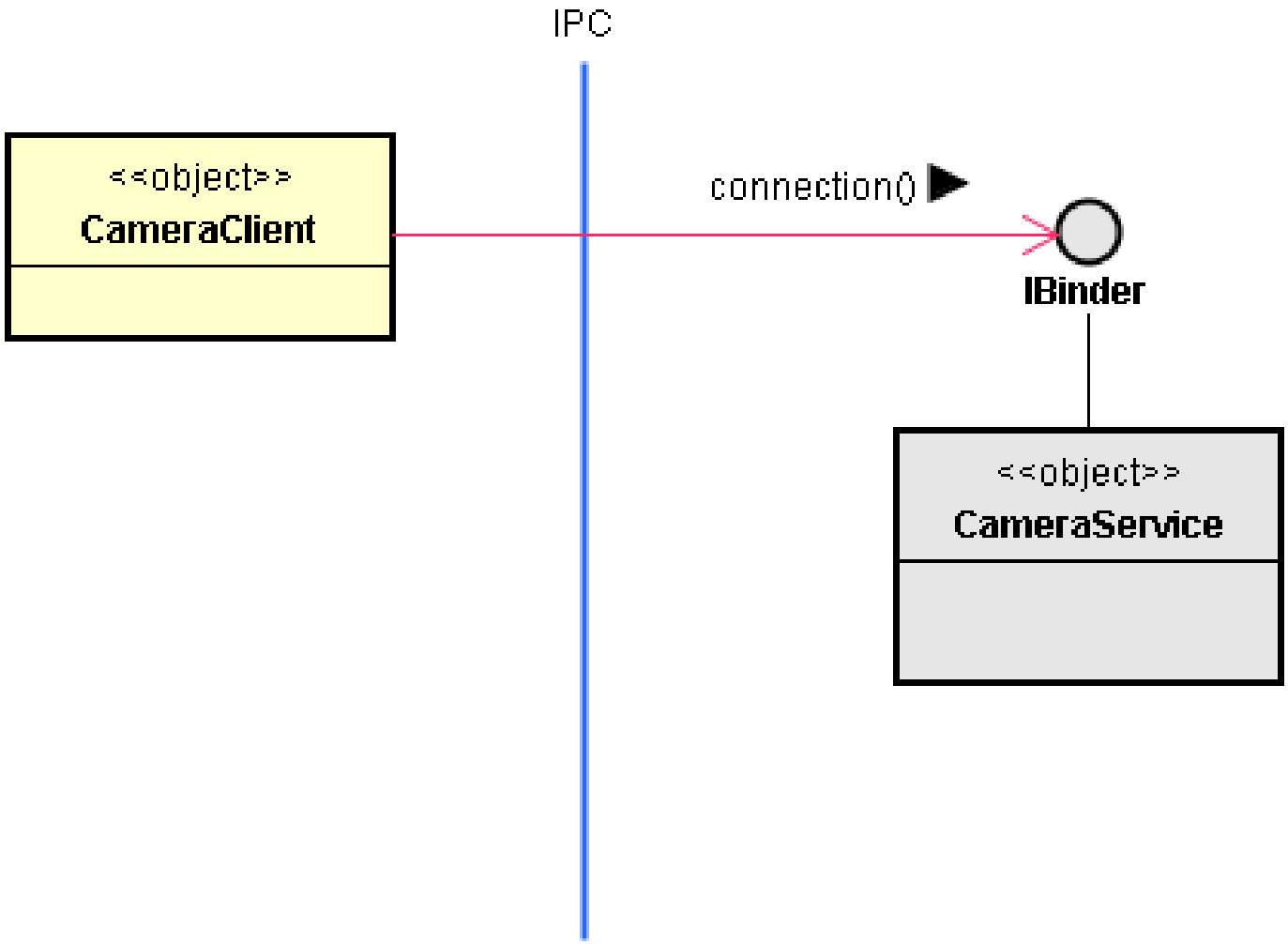
- 当很多个应用程序来衔接CameraService时，该如何接待呢?


IPC
<<object>>
CameraClient
<<object>>
CameraClient
<<object>>
CameraClient
IBinder
<<object>>
CameraService

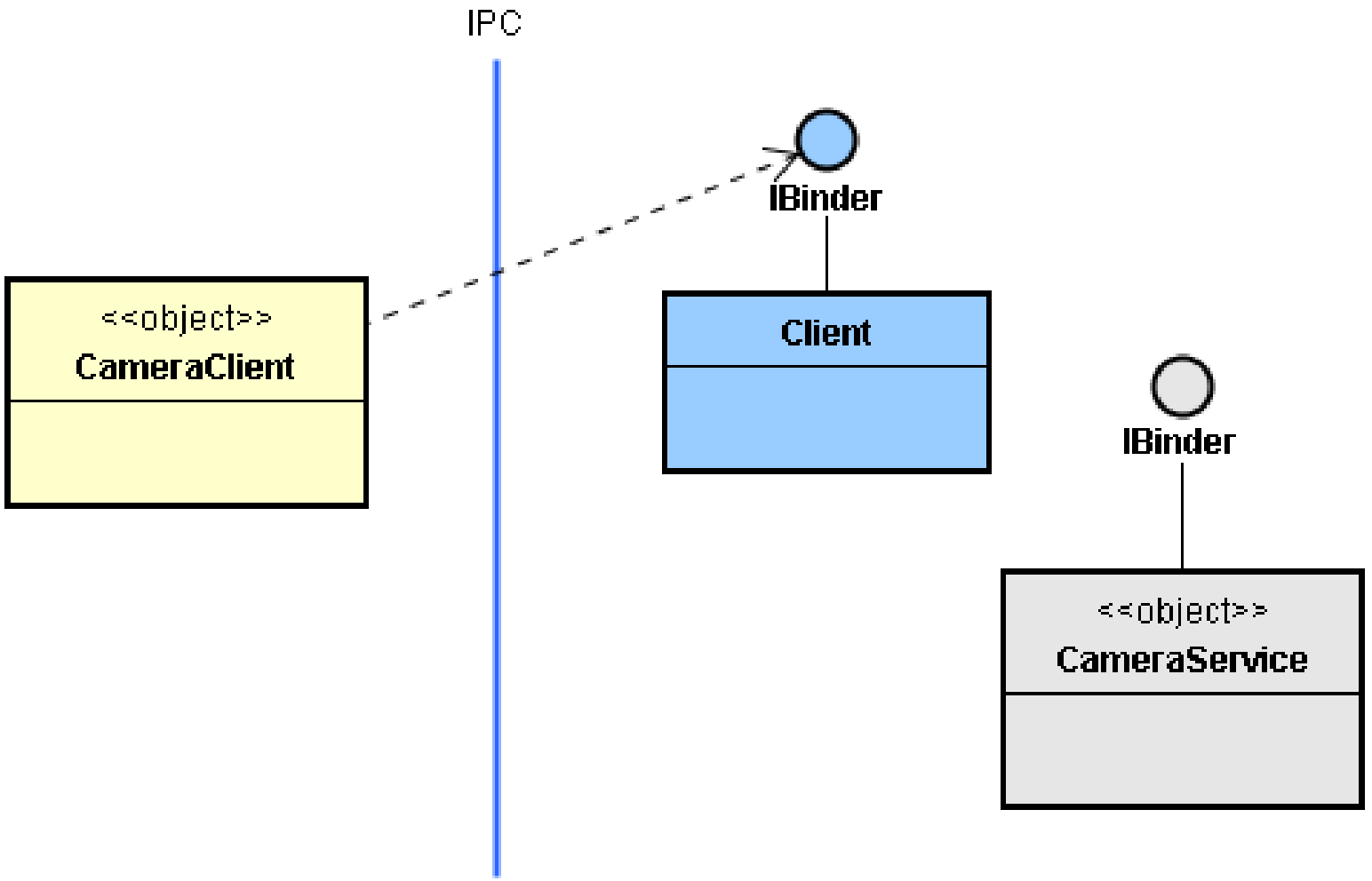

IPC
<<object>>
CameraClient
<<object>>
CameraClient
<<object>>
CameraClient
IBinder
<<object>>
CameraService

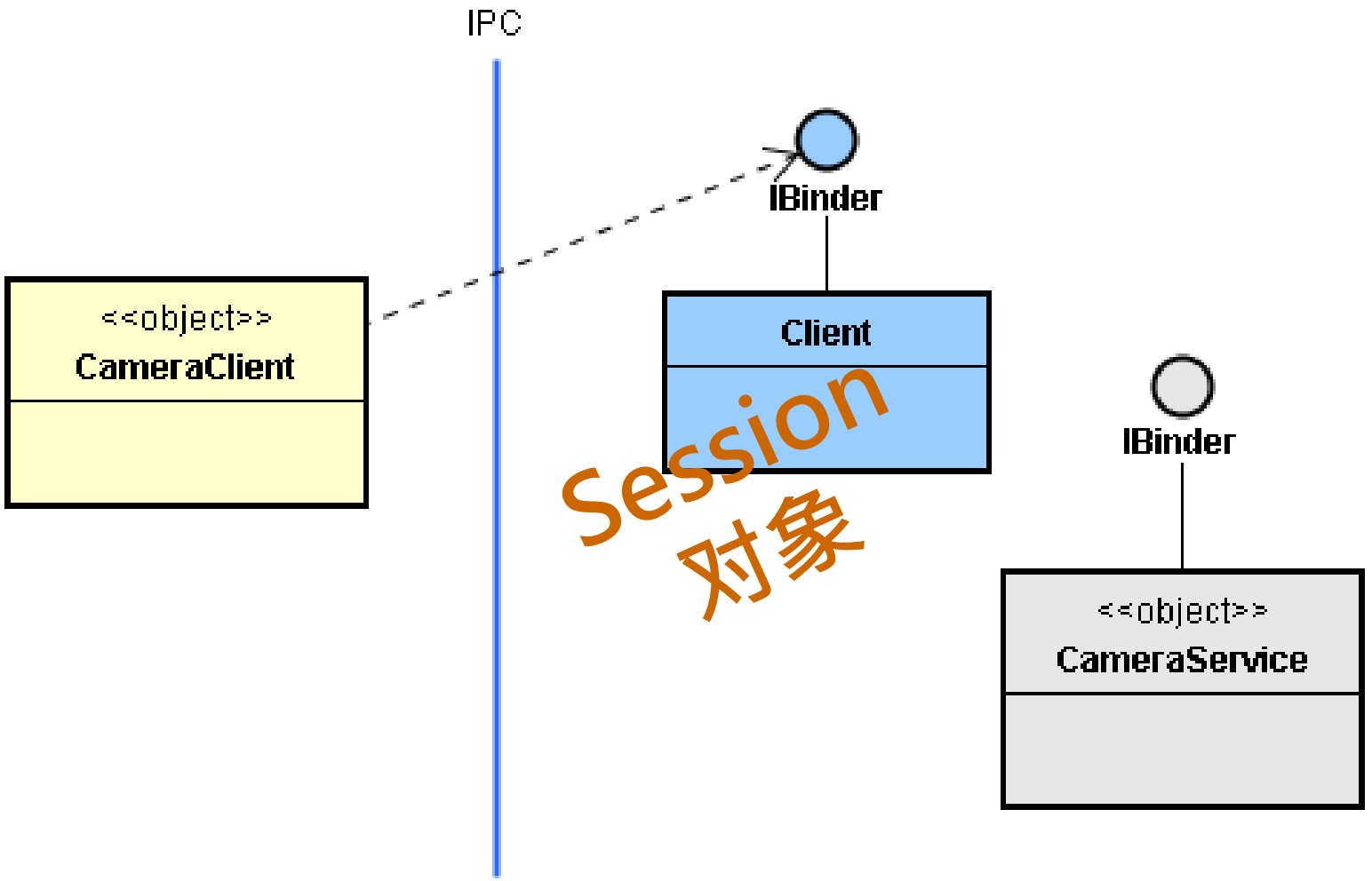
采用Session设计模式

- 步骤一、CameraClient先调用CameraService，这个动作通称为：取得取得连线 (Connection)。

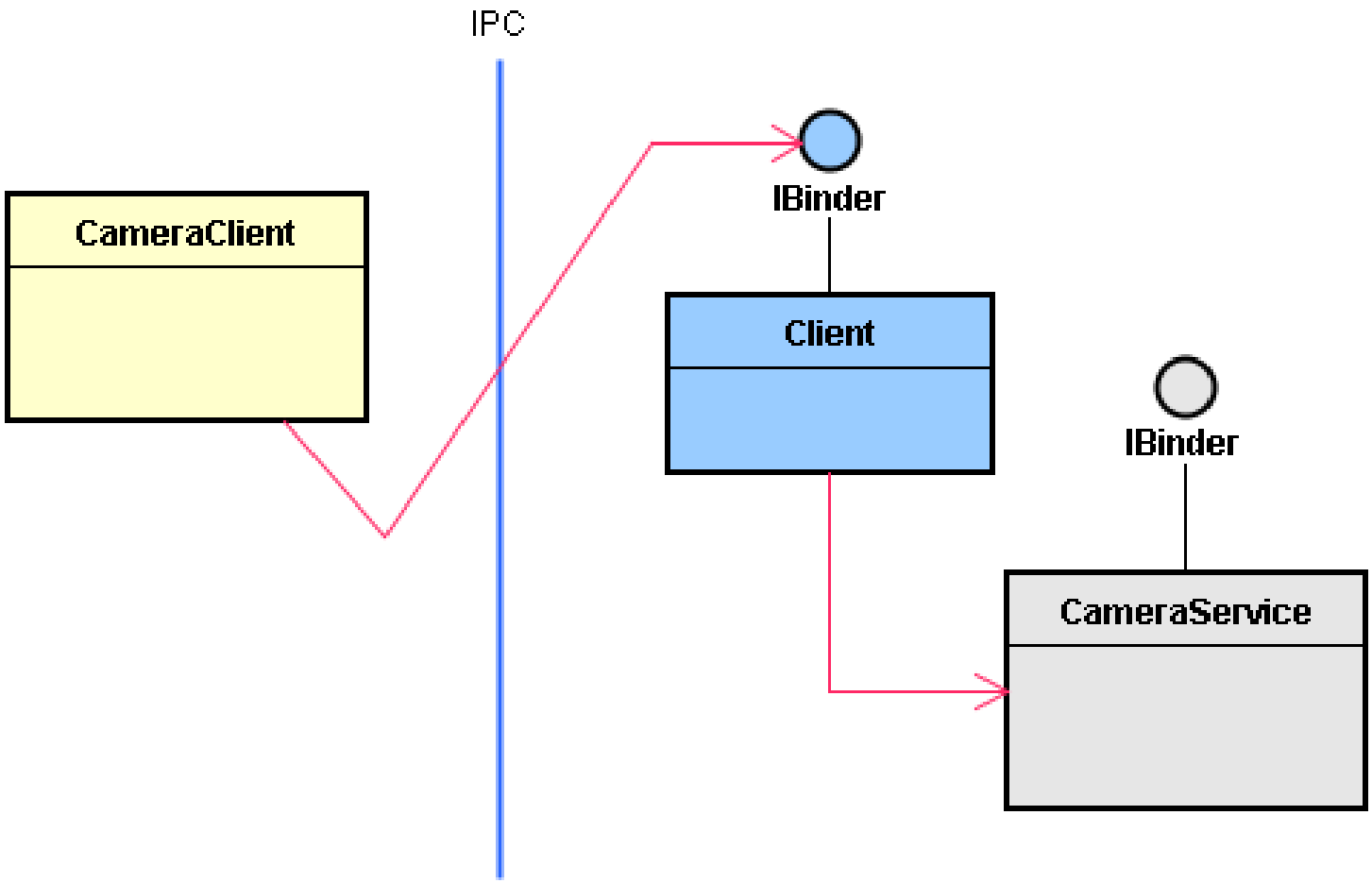

IPC
<<object>>
CameraClient
connection()
IBinder
<<object>>
CameraService

- 此时，CameraService创建一个Client对象，并将Client的IBinder接口回传给CameraClient。

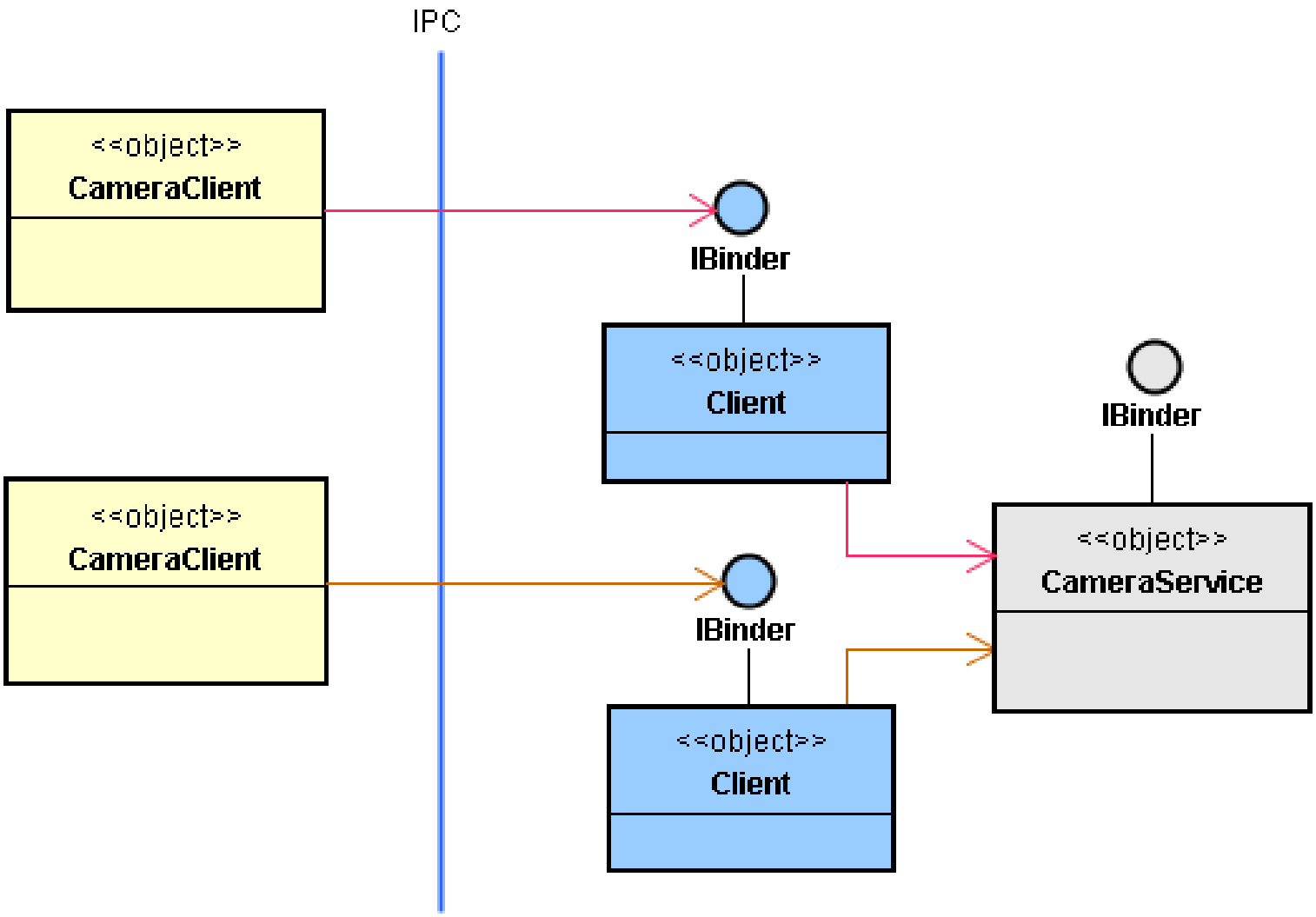

IPC
IBinder
<<object>>
CameraClient
Client
IBinder
<<object>>
CameraService


IPC
IBinder
<<object>>
CameraClient
Client
Session
对象
IBinder
<<object>>
CameraService

- 步骤二、CameraClient就调用Client，要求Client与CameraService通信，间接调用到CameraService的服务。


IPC
CameraClient
IBinder
Client
IBinder
CameraService

每一个Connection
都有一个私有的Session对象


IPC
<<object>>
CameraClient
IBinder
<<object>>
Client
IBinder
<<object>>
CameraClient
IBinder
<<object>>
CameraService
<<object>>
Client

- 当CemeraClient透过SM绑定了CameraService之后，Camera就可调用CameraService的connect()函数来建立一个连线。
- 这时，CameraClient把自己的<ICameraClient>接口传递给CameraService。
- 例如下述的Android程序源码：

```
sp<ICamera> CameraService::connect(const sp<ICameraClient>&
cameraClient)
{
    Mutex::Autolock lock(mLock);
    sp<Client> client;
    if (mClient != 0) {
        // 這個cameraClient已經調用過connect()了
        // 先前已經給它一個Client對象(mClient)了
        // 就先將mClient轉型態
        sp<Client> currentClient = mClient.promote();
        // 這currentClient就是mClient
        if (currentClient != 0) {
            sp<ICameraClient> currentCameraClient(
                            currentClient->getCameraClient());
```

```
        // 比對一下mClient所記錄的cameraClient與
        // 本次來訪的cameraClient是否同一個
        if (cameraClient->asBinder()
                == currentCameraClient->asBinder()) {
            return currentClient;
            // 如果同一個，回傳先前的Client對象
        } else  return client;
      } else  mClient.clear();
    }
    // 這個cameraClient是第一次來訪
    // 建立一個新Client對象
    client = new Client(this, cameraClient,
                        IPCThreadState::self()->getCallingPid());
    mClient = client;
    return client;
}
```

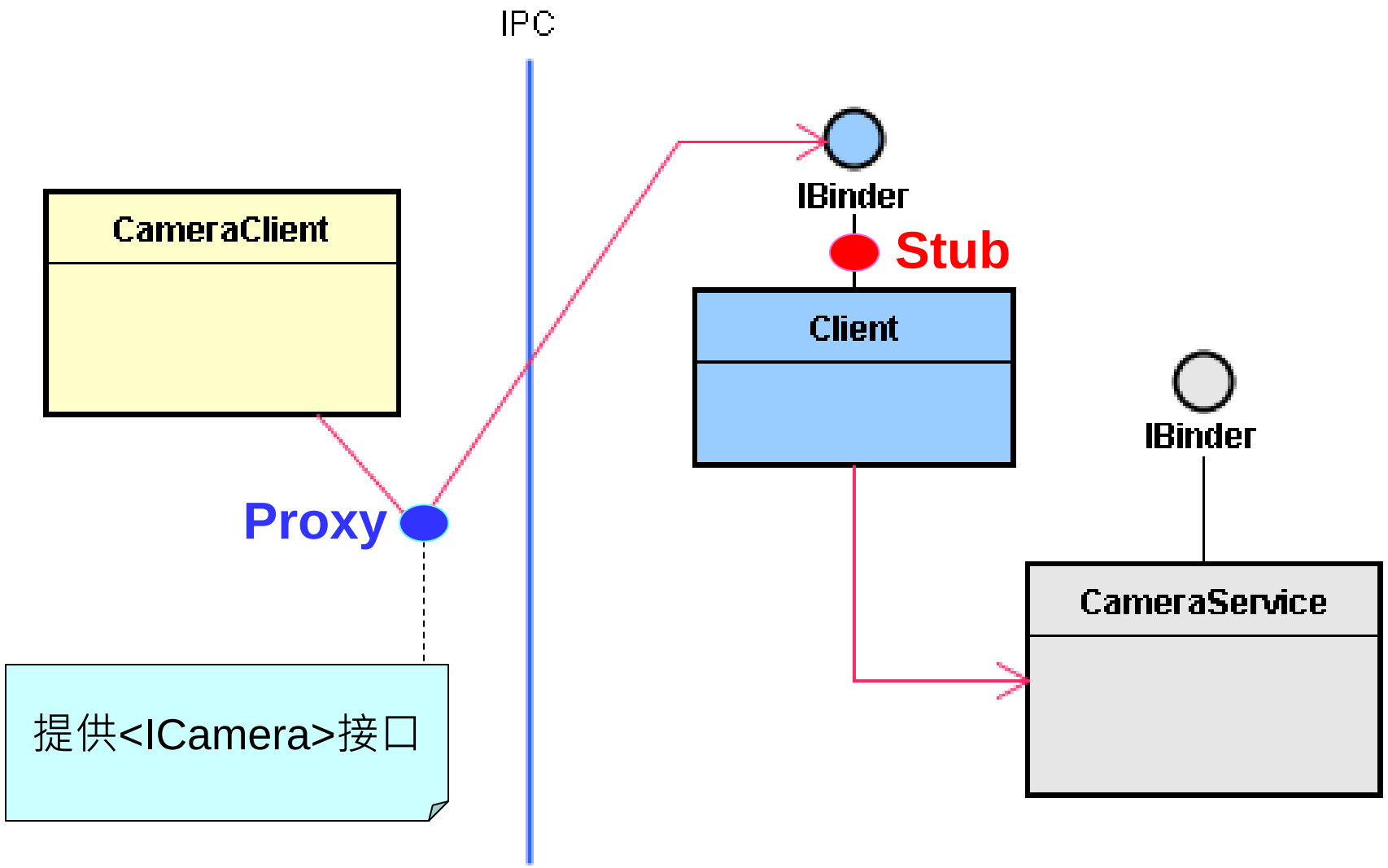
- 这时，CameraService就在自己进程里诞生了一个Client的对象。
- 并且，将此Client对象的<ICamera>接口回传给CameraClient。

```
sp<ICamera> CameraService::connect( const
                        sp<ICameraClient>& cameraClient)
{
    // .........
}
```

那么，
<ICamera>接口和
<ICameraClient>接口，
从那里来的呢? 定义在那里呢?

答案是：
来自Proxy-Stub设计模式。

也就是BnInterface<T>和BpInterface<T>模板所建立的Proxy和Stub类。


IPC
IBinder
Stub
CameraClient
Client
IBinder
Proxy
CameraService
提供<ICamera>接口

- 待会儿，来介绍Session模式与Proxy-Stub模式的美妙搭配。

~ Continued ~