20180403_Windows和Linux进程间通信区别

```
windows 中 管道分为匿名和命名，
Linux中 消息，工作，访问模式

windows 中有油槽： 一个对多个发（广播）
Linux中： 消息队列，多个对一个发

Socket 通信：不同机器之间的通信
```

多进程和多线程本质上就是将原来一个进程或者线程处理的任务分给了多个进程或者线程，也可以说是将原来一个CPU处理的任务分给了多个CPU处理，类似于随着生产力的发展，原来一个人包打天下的个人英雄主义时代被分工合作的团队取代一样。

既然是一个团队，就必然涉及到**分工合作**问题，并行程序的设计本质上就是解决“分工”和“合作”的问题。其中“分工”主要是后面讲到“并行程序设计模式”，而“合作”则是本篇重点要讨论的问题。

相信大家做过项目的都知道所谓合作其实就是**“通信”和“冲突解决”**。常见的“通信”有开会、电话、邮件、甚至QQ等，无非是为了交换信息，而“冲突解决”其实就是为了解决资源不足，大家抢着用的问题。

可能许多菜鸟、大侠以及网上的很多博客都和和我开始一样，将“通信”和“冲突解决”混为一谈了，一说到进程间通信，就会口若悬河的冒出一大堆名词：管道、信号量、消息队列、互斥。。。。。这里面有的是“通信”（**管道、消息队列）**，有的是“冲突解决”（信号量、互斥），并不是一类东东。

下面我们分别从“通信”和“冲突解决”两方面来比较Windows和Linux。

1.1 Windows进程间通信

标准（例如《Windows系统编程》里面提到的）的Windows进程间通信有三个：匿名管道、命名管道（又叫FIFO）、邮槽（MailSlot），实际上常用的还有一个：共享内存。之所以说它不是标准的，我猜测可能是共享内存设计本意不是为了进程间通信用的，而是为了内存映射用的。

1.1.1 匿名管道

顾名思义：匿名管道就是“匿名”的“管道”。为什么这样拆开呢？正所谓名如其人，通过名字我们就可以了解大概这是个什么东东。

匿名：之所以叫做“匿名”，当然是因为没有名字了，但为什么会没有名字呢？没有名字又有什么好处呢？其实很简单了，“匿名”当然是不想让其它人知道了，说白了这个“匿名管道”**就是只给父子进程用的，别人不需要也不可能知道名字的。**

管道：说道管道，你是否想到了“下水管道”、“煤气管道”等？对了，和这些管道本质上是一样的，就是可以传送东西的，所以叫做管道。要注意匿名管道是**“单向流通”（也叫半双工）的。**

1.1.2 命名管道

聪明的你看到这个名字肯定就会产生如下两个想法，我们就一一来解答：

1）和匿名管道看起来很像

是的，命名管道就是相对匿名管道来说的。

2）命名管道和匿名管道有什么差别？

对比点	匿名管道	命名管道	备注
消息格式	字符	二进制	命名管道可以控制读消息的长度
工作模式	半双工	全双工	NA
访问模式	只能在一台机器上	可以跨网络	NA
通信模式	一对一，父子进程用	一对多，不同的进程都可以用	一个命名管道可以有多个实例

1.1.3 邮槽

邮槽和命名管道类似，都是有名字的，也可以跨网络进行通信，既然是这样，为什么Windows还要设计邮槽呢？其实也没有什么玄虚，说简单点就是管道都是"点对点"的（命名管道虽然是一对多，但具体的通信还是1对1进行的），而**邮槽是为了提供一种"广播"通信机制（王婆卖瓜一下：微软还不如将邮槽叫做"广播"：-P）。**

下面我们看看邮槽和命名管道的对比：

对比点	邮槽	命名管道	备注
消息格式	数据包	二进制	命名管道可以控制读消息的长度
工作模式	单向	全双工	广播当然是单向的了
访问模式	可以跨网络	可以跨网络	NA

1.1.4 共享内存

就像前面提到的一样，共享内存并不是正统的进程间通信的机制，共享内存其实只不过是Windows"内存映射文件"的一个特殊用法而已。

然而实际中共享内存在进程间通信却比较常见，从使用方便性上来说，共享内存其实没有前面介绍的方便（必须结合互斥、事件等一起使用），但为什么应用比较多呢？关键在于**共享内存性能很高。**

共享内存应该是**介于匿名管道和命名管道之间的通信方式**，为什么这么说呢？主要有如下几个原因：

1）共享内存和匿名管道相比：共享内存有名称，可用于多个进程通信，这点像命名管道；

2）共享内存和命名管道相比：共享内存只能在一台机器上使用，不能跨网络，这点和匿名管道类似

3）共享内存是双向的，这点又和命名管道类似。

1.2 Linux进程间通信

介绍完了Windows，介绍Linux就相对轻松一些了，虽然Windows和Linux形同水火，打的不可开交，但实际上说白了，它们并不是两个完全不同的东东，在很多的地方都相似，进程间通信也不例外。

Linux的进程间通信主要有**管道、命名管道、消息队列、共享内存、信号量，**其中信号量Semaphore其实是为了同步用的，因此我这里就放到下一篇关于同步的博文中去分析。另外，很多人将信号signal也作为进程间通信，但我认为信号更像是为了同步而设计的，因此也放到下一篇博文中去分析。

1.2.1 管道

Linux的管道和Windows的管道是一样的，这里就不详细介绍了。

需要注意的是**Linux多了一个叫做"流管道"的东东**，除了流管道是全双工（也就是双向）外，流管道其它都和管道一样。

1.2.2 命名管道

Linux的命名管道和Windows的**命名管道差异就比较大了**，主要对比如下：

对比点	Linux	Windows	备注
消息格式	字节流	二进制	Windows更牛
工作模式	半双工	全双工	Windows更牛
访问模式	只能在一台机器上	可以跨网络	Windows更牛

1.2.3 消息队列

这个是Linux特有的进程通信方式，我感觉它有点像邮槽，都能够实现一对多。不过消息队列和邮槽的方向正好相反：**消息队列是一堆进程向一个进程发，邮槽是一个进程向一堆进程发。**

消息队列还有一个牛B的特性就是**消息队列的消息可以分优先级，进程不一定非要取第一个消息，也可以取指定消息优先级的消息。**因此这个特性又可用于多对多通信，即：不同的收方指定不同的优先级。当然实际应用中应该没人会这么用，直接创建多个消息队列是最方便、最简单、效率最高的方法。

1.2.4 共享内存

Linux的共享内存机制和Windows本质上是一样的，即都是利用了内存映射的功能。不过**Linux将"内存映射到内存"包装成了"共享内存"，而不像Windows，在使用的时候通过指定不同的参数来区分是"内存映射到文件"还是"共享内存"，**所以各位大侠可能在网上有时能够看到有人将**"内存映射"和"共享内存"**都说是Linux进程间通信的方式，原因就在这里。

1.3 OS间进程通信

前面分别介绍了Windows和Linux进程间通信，看到这里，你肯定会有疑问：什么？还有OS进程间通信？不同OS之间的进程是不可能通信的！

是的，从操作系统层面来说，Windows的进程和Linux的进程当然是不能通信的，但如果从网络层面来说，两台机器总是要通信的吧？不可能Windows只能和Windows通信，Linux只和Linux通信吧？

说到这里估计你已经恍然大悟了：不就是**Socket通信么？**

是的，就是它，虽然它是用于机器间通信的，但大家想想，机器间通信不就是各个应用程序通信么？各个应用程序不就是对应操作系统中的一个或者多个进程么？