<https://blog.csdn.net/wujiafei_njgcxy/article/details/77098977>

1.什么是进程？什么是线程？

进程是表示资源分配的基本单位，又是调度运行的基本单位。例如，用户运行自己的程序，系统就创建一个进程，并为它分配资源，包括各种表格、内存空间、磁盘空间、I/O设备等。然后，把该进程放人进程的就绪队列。进程调度程序选中它，为它分配CPU以及其它有关资源，该进程才真正运行。所以，进程是系统中的并发执行的单位。  
在Mac、Windows NT等采用微内核结构的操作系统中，进程的功能发生了变化：它只是资源分配的单位，而不再是调度运行的单位。在微内核系统中，真正调度运行的基本单位是线程。因此，实现并发功能的单位是线程。  
线程概念  
　　线程是进程中执行运算的最小单位，亦即执行处理机调度的基本单位。如果把进程理解为在逻辑上操作系统所完成的任务，那么线程表示完成该任务的许多可能的子任务之一。例如，假设用户启动了一个窗口中的数据库应用程序，操作系统就将对数据库的调用表示为一个进程。假设用户要从数据库中产生一份工资单报表，并传到一个文件中，这是一个子任务；在产生工资单报表的过程中，用户又可以输人数据库查询请求，这又是一个子任务。这样，操作系统则把每一个请求――工资单报表和新输人的数据查询表示为数据库进程中的独立的线程。线程可以在处理器上独立调度执行，这样，在多处理器环境下就允许几个线程各自在单独处理器上进行。操作系统提供线程就是为了方便而有效地实现这种并发性。

 举个例子来说多线程就像是火车上的每节车厢，而进程就是火车。

2.多进程和多线程的区别？

我们从各个方面来看待这个问题，由下面的图片说明：

|  |  |  |  |
| --- | --- | --- | --- |
| **对比维度** | **多进程** | **多线程** | **总结** |
| 数据共享、同步 | 数据共享复杂，需要用IPC；数据是分开的，同步简单 | 因为共享进程数据，数据共享简单，但也是因为这个原因导致同步复杂 | 各有优势 |
| 内存、CPU | 占用内存多，切换复杂，CPU利用率低 | 占用内存少，切换简单，CPU利用率高 | 线程占优 |
| 创建销毁、切换 | 创建销毁、切换复杂，速度慢 | 创建销毁、切换简单，速度很快 | 线程占优 |
| 编程、调试 | 编程简单，调试简单 | 编程复杂，调试复杂 | 进程占优 |
| 可靠性 | 进程间不会互相影响 | 一个线程挂掉将导致整个进程挂掉 | 进程占优 |
| 分布式 | 适应于多核、多机分布式；如果一台机器不够，扩展到多台机器比较简单 | 适应于多核分布式 | 进程占优 |

3.进程之间的通信方式以及优缺点？

1)管道  
管道分为有名管道和无名管道  
无名管道是一种半双工的通信方式,数据只能单向流动,而且只能在具有亲缘关系的进程间使用.进程的亲缘关系一般指的是父子关系。无明管道一般用于两个不同进程之间的通信。当一个进程创建了一个管道,并调用fork创建自己的一个子进程后,父进程关闭读管道端,子进程关闭写管道端,这样提供了两个进程之间数据流动的一种方式。  
有名管道也是一种半双工的通信方式,但是它允许无亲缘关系进程间的通信。  
  
无名管道：优点：简单方便；缺点：1）局限于单向通信2）只能创建在它的进程以及其有亲缘关系的进程之间;3）缓冲区有限；  
有名管道：优点：可以实现任意关系的进程间的通信；缺点：1）长期存于系统中，使用不当容易出错；2）缓冲区有限  
2)信号量  
信号量是一个计数器,可以用来控制多个线程对共享资源的访问.,它不是用于交换大批数据,而用于多线程之间的同步.它常作为一种锁机制,防止某进程在访问资源时其它进程也访问该资源.因此,主要作为进程间以及同一个进程内不同线程之间的同步手段.  
优点：可以同步进程；缺点：信号量有限  
  
3)信号  
信号是一种比较复杂的通信方式,用于通知接收进程某个事件已经发生.  
  
4)消息队列  
消息队列是消息的链表,存放在内核中并由消息队列标识符标识.消息队列克服了信号传递信息少,管道只能承载无格式字节流以及缓冲区大小受限等特点.消息队列是UNIX下不同进程之间可实现共享资源的一种机制,UNIX允许不同进程将格式化的数据流以消息队列形式发送给任意进程.对消息队列具有操作权限的进程都可以使用msget完成对消息队列的操作控制.通过使用消息类型,进程可以按任何顺序读信息,或为消息安排优先级顺序.  
  
优点：可以实现任意进程间的通信，并通过系统调用函数来实现消息发送和接收之间的同步，无需考虑同步问题，方便；缺点：信息的复制需要额外消耗CPU的时间，不适宜于信息量大或操作频繁的场合  
  
5)共享内存  
共享内存就是映射一段能被其他进程所访问的内存,这段共享内存由一个进程创建,但多个进程都可以访问.共享内存是最快的IPC(进程间通信)方式,它是针对其它进程间通信方式运行效率低而专门设计的.它往往与其他通信机制,如信号量,配合使用,来实现进程间的同步与通信.  
  
优点：无须复制，快捷，信息量大；缺点：1）通信是通过将共无法实现享空间缓冲区直接附加到进程的虚拟地址空间中来实现的，因此进程间的读写操作的同步问题；2)利用内存缓冲区直接交换信息，内存的实体存在于计算机中，只能同一个计算机系统中的诸多进程共享，不方便网络通信  
  
6)套接字：可用于不同及其间的进程通信  
优点：1）传输数据为字节级，传输数据可自定义，数据量小效率高；2）传输数据时间短，性能高；3) 适合于客户端和服务器端之间信息实时交互；4) 可以加密,数据安全性强  
缺点：1) 需对传输的数据进行解析，转化成应用级的数据。

4.线程之间的通信方式？

 # 锁机制：包括互斥锁、条件变量、读写锁  
   \*互斥锁提供了以排他方式防止数据结构被并发修改的方法。  
   \*读写锁允许多个线程同时读共享数据，而对写操作是互斥的。  
   \*条件变量可以以原子的方式阻塞进程，直到某个特定条件为真为止。对条件的测试是在互斥锁的保护下进行的。条件变量始终与互斥锁一起使用。  
# 信号量机制(Semaphore)：包括无名线程信号量和命名线程信号量  
# 信号机制(Signal)：类似进程间的信号处理  
    线程间的通信目的主要是用于线程同步，所以线程没有像进程通信中的用于数据交换的通信机制。

5.什么时候用多线程?什么时候用多进程？

1）需要频繁创建销毁的优先用线程  
原因请看上面的对比。  
这种原则最常见的应用就是Web服务器了，来一个连接建立一个线程，断了就销毁线程，要是用进程，创建和销毁的代价是很难承受的  
2）需要进行大量计算的优先使用线程  
所谓大量计算，当然就是要耗费很多CPU，切换频繁了，这种情况下线程是最合适的。  
这种原则最常见的是图像处理、算法处理。  
3）强相关的处理用线程，弱相关的处理用进程  
什么叫强相关、弱相关？理论上很难定义，给个简单的例子就明白了。  
一般的Server需要完成如下任务：消息收发、消息处理。“消息收发”和“消息处理”就是弱相关的任务，而“消息处理”里面可能又分为“消息解码”、“业务处理”，这两个任务相对来说相关性就要强多了。因此“消息收发”和“消息处理”可以分进程设计，“消息解码”、“业务处理”可以分线程设计。  
当然这种划分方式不是一成不变的，也可以根据实际情况进行调整。  
4）可能要扩展到多机分布的用进程，多核分布的用线程  
原因请看上面对比。  
5）都满足需求的情况下，用你最熟悉、最拿手的方式  
至于“数据共享、同步”、“编程、调试”、“可靠性”这几个维度的所谓的“复杂、简单”应该怎么取舍，我只能说：没有明确的选择方法。但我可以告诉你一个选择原则：如果多进程和多线程都能够满足要求，那么选择你最熟悉、最拿手的那个。

<https://blog.csdn.net/jiazhen/article/details/1611721> 线程间的通讯

<https://www.cnblogs.com/LUO77/p/5816326.html> 进程间的通信

