同步进程设计及应用

王建民 任文科 （建设银行安阳分行 455000）

友好的用户界面设计是评价一个应用系统的重要指标。在 UNIX 操作系统环境下，对大型数据库的操作或网络上与其他节点交换信息时，需要一定的执行时间。对用户来说，应用程序的运行是不可见的。若用户界面做得欠佳，程序执行时，屏幕上没有动态提示，易给用户产生“死机”的错觉。我们在实际工作中，利用 UNIX 操作系统中同步进程设计技术，很好的解决了这个问题，收到了良好的效果。

一、同步进程设计

UNIX 系统中，信号量（semcephores）为我们提供了一种实现进程同步的通信方法。也就是说我们可以通过进程间信号量的传递，达到两个进程同步运行。下面概括介绍一下同步进程设计中用到的信号及系统调用：

1. 信号。UNIX 为我们提供了多种信号类型，标准包含文件＜signal. h＞中对这些信号都做了定义。SIGUSR1 和 SIGUSR2 这两种信号为用户定义信号，可用于用户所希望的任何目的。

2. 系统调用 signal 能把指定的信号和指定的处理工作联系起来，它的使用格式如下：

signal(sig, func);

signal 语句执行后，进程只要接收到一个指定的信号 sig，不管其正在执行程序的哪一部分，都会立即执行指定响应的动作 func。当 func 返回时，控制被传回到进程被中断的那一点，继续执行。例如：

signal(SIGUSR1, ac1);

就表示进程只要一接到信号 SIGUSR1，就立即执行函数 ac1。

3. 系统调用 kill 用来将一指定信号发送给一指定进程。它的使用格式如下：

kill(pid, sig);

由于调用 kill 的进程需要知道信号发往之进程的进程标识符 PID，因此，这种信号发送通常在父进程、子进程之间进行。例如：

kill(ppid, SIGUSR1);

这一语句用来实现把信号 SIGUSR1 发送给进程标识符为 ppid 的进程。

4. 系统调用 fork 可产生一个与源进程相同的子进程（子进程继承父进程所有图象内容），并响应子进程的 PID 给父进程，而响应 0 给子进程。每当子进程产生后，即用下一个可用的进程标识符赋予子进程。

下面以一个简单的例子，介绍一下同步进程设计技术。该例子通过系统调用 fork 形成父、子两个进程。fork 响应 0 给子进程，响应子进程的 PID 给父进程，因此子进程执行 switch 开关的“case 0”语句中的循环，父进程执行“default”语句中的循环。子、父进程各自通过 kill 调用向对方发送信号 SIGUSR1，以实现它们之间的同步。系统调用 pause 使一个进程暂停，直至一个信号到达。父、子进程收到信号 SIGUSR1 后，便立即执行函数 ac1、ac2，实现同步向各自的标准输出上写信息。

例子一：

```
#include<stdio.h>
#include<signal.h>
int ntime = 0 ;
main()
{
    int pid, ppid;
    int ac1(), ac2();
    signal(SIGUSR1, ac1);
    ppid = getpid();
    switch(pid = fork()){
      case -1 :
        perror("synchro");
        exit(1);
      case 0:
        signal(SIGUSR1, ac2);
        for(;;){
          sleep(1);
          kill(ppid, SIGUSR1);
          pause();
        }
```

```
        default:
          for(;;){
            pause();
            sleep(1);
            kill(pid,SIGUSR1);
            }
      }
}

ac1()
      {
        printf("parent caught signal %d \n", ++ntime);
        signal(SIGUSR1,ac1);
      }
ac2()
      {
        printf("child caught signal %d \n", ++ntime);
        signal(SIGUSR1,ac2);
      }
```

该程序的运行结果如下：

```
parent caught signal 1
child caught signal 1
parent caught signal 2
child caught signal 2
<del>（用户按中断键退出）
```

二、进程同步设计技术的应用

建设银行城市综合网络采用了 client/server 模式。所有业务数据均集中 server 中，前置机中只有一些标准数据。办理业务时，client 端向 server 发送请求信息，server 根据请求信息启动相应进程运行，并将运行的结果信息反馈给前置机。前置机根据 server 的回应信息作出相应处理。在前置机和 server 交换数据时，前置机进程处于等待回应状态，而数据交换需要一定的时间。此时，前置机标准输出上如无任何动态提示，易给用户产生“死机”错觉。为解决这个问题，我们利用 unix 系统的同步进程设计技术，在父进程与 server 交换信息时，启动子进程在标准输出上打印出各种动态提示信息，以表示此笔业务正在处理，给用户以明确的提示。下面这个程序段实现了父进程在执行函数 sjjh()时，启动子进程不停的在标准输出上打印时间数字，以表示父进程正在与 server 交换数据。当函数 sjjh()执行结束后，父进程将发送信号 SIGUSR1 到子进程，子进程接到信号后，立即执行函数 childkill，子进程被终止，父进程继续向下执行。

例子二：

```
            ……
main()
    {
            ……
      void childkill()
            ……
      switch ( chpid = fork() )
      {
        case -1:
          “出错处理函数”
          break;
        case 0:
          signal ( SIGUSR1, childkill );
          while (1)
          {
            times++;
            printf("%d", times);
            sleep(1);
        /*在标准输出上不断给出动态提示信息*/
          }
        default:
          sjjh(); /* 数据交换函数 */
          kill ( chpid, SIGUSR1 );
          break;
      }
          ……
          ……
    }
    void childkill()
    {
      exit(0);
    }
```

三、结束语

通过信号量传递实现进程同步执行的设计技术，结构简单，通用性强。我们在建设银行城市综合网络应用程序开发与维护中，普遍采用了这种进程同步设计技术，使得该系统的界面设计显得十分生动、活泼，取得了良好的效果。

（来稿时间：1997 年 12 月）