第33卷 第13期 计 算 机 工 程 2007年7月
Vol.33 No.13 Computer Engineering July 2007
·工程应用技术与实现· 文章编号：1000—3428(2007)13—0255—02 文献标识码：A 中图分类号：TP39

基于 UDP 协议的群发控制系统设计与实现

石光华
(深圳职业技术学院计算机系，深圳 518055)

摘 要：针对如何解决局域网环境下多进程群发控制问题，提出了采用自定义的 ATCP 协议进行客户端多进程群发控制的方案。通过利用 UDP 协议，采用进程枚举和比对的方法，实现了 LAN 环境下多机多进程的实时群发控制，对该方案的实际运用效果进行了评价。
关键词：枚举进程；广播；协议

Group Control Method of Multi-process Based on UDP

SHI Guanghua
(Department of Computer, Shenzhen Polytechnic College, Shenzhen 518055)

【Abstract】This paper gives a group control method of multi-process in a LAN. Based on UDP protocl, using enumerate processes and compare multi-process, through the definition of ATCP (assistant teaching control protocol)agreement, real-time sends the agreement and automatic controls the user processes of a group clients. The performance of this method is estimated through test data.
【Key words】enumerate processes; broadcast; protocol

在计算机机房授课时，除了教学必需的程序外，学生可以随意运行与教学无关的程序，严重地影响了教学秩序。针对这个问题笔者提出了一种切实可行的解决办法：以服务器设定的进程数据为基准，客户端自动监控学生使用的进程，实现了“让你动，但不让你乱动”的监控目标，解决了“要么不让动，让动就乱动”问题，只监视非法进程，不影响正常操作，避免屏幕监视造成的逆反心理，真正成为教师的教学助手。实际测试和应用已经达到了实用要求。

系统主要功能模块包括进程管理模块、鼠标/键盘控制模块和辅助管理模块。其中，进程管理模块是系统的核心，采用客户/服务器模式。教师使用服务器，设定允许学生运行的程序；学生使用客户端时，只能运行许可的程序，而其他程序的运行则会被自动制止。

1 系统结构

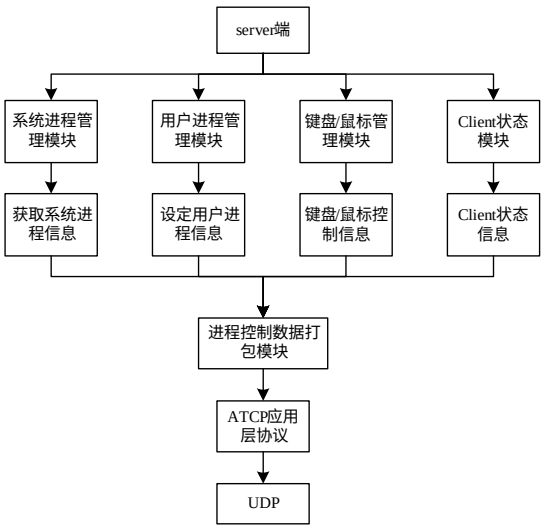

图 1 Server 端系统结构和数据流程

系统主要功能模块包括进程管理模块、鼠标/键盘控制模块和辅助管理模块。系统采用 Server/Client 结构，Server 端为教师使用，Client 端为学生使用。在 Client 端允许运行的软件进程称为合法进程，在 Server 端可以设定合法进程，并通过 UDP 协议广播至局域网，则局域网上所有 Client 端均能收到合法进程的报文，Client 端根据报文内容，与当前系统中运行的进程快照进行比对，从而实现 Client 端只能运行许可的程序，而其他程序则会自动制止其运行的控制目标。Server 端系统结构和数据流如图 1 所示。

Server 端的控制信息通过自定义的协议打包后，通过 UDP 协议在局域网内广播，如图 1 所示。Client 端收到广播后，按图 2 所示的流程解包，关闭本机上运行的非法程序。ATCP(assistant teaching control protocol)是自定义的辅助教学控制协议，作用是在 Server 和 Client 间传递控制数据包。

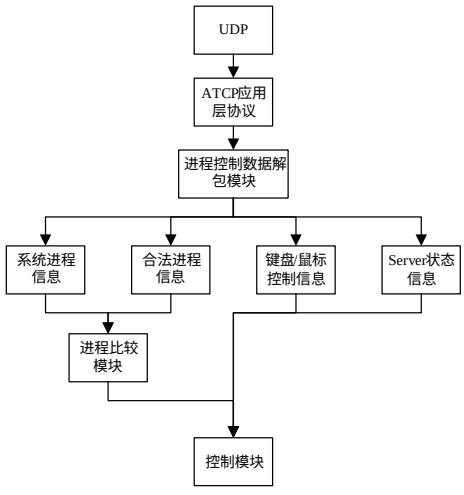

图 2 Client 端结构和数据流程

作者简介：石光华(1962－)，男，讲师、硕士，主研方向：计算机网络技术，软件工程，项目管理
收稿日期：2006-11-02 **E-mail：**stone688@szpt.net

2 关键技术

2.1 进程快照的获取

通过获取某一时刻系统的进程快照，与合法进程信息进行比对，找出 Client 端运行的非法进程并立即关闭它，是进行多用户多进程自动管理的关键。由于 Windows NT 处理方法的特殊，为了保证在所有 32 位 Windows 平台上都能取得进程快照，采用两种方法来获得进程快照：使用 ToolHelp32 库枚举进程和使用 PSAPI 库枚举进程。

2.1.1 使用 ToolHelp32 库枚举进程

在 Windows 9x、Windows ME、Windows 2000 Professional 以及 Windows XP 中，可以用 ToolHelp32 库中的 APIs 函数获取 32 位程序的进程快照。使用 ToolHelp32 库枚举进程只需用如下 3 个函数：CreateToolhelp32Snapshot()，Process32First() 和 Process32Next()。首先用 CreateToolhelp32Snapshot()创建进程链表的句柄，然后先调用一次 Process32First()，再调用 Process32Next()，直到其中一个函数返回 FALSE 为止。在这个过程中，进程的所有信息被放到进程结构 PROCESSENTRY32 中。流程如图 3 所示。结构 PROCESSENTRY32 的定义如下：

```
typedef struct tagPROCESSENTRY32 {
DWORD dwSize;//本结构的大小。必须在使用前初始化为
//sizeof(PROCESSENTRY32).
    DWORD cntUsage;//不再使用，一直为 0
    DWORD th32ProcessID;//进程号 ID
    ULONG_PTR th32DefaultHeapID;//不再使用，一直为 0
    DWORD th32ModuleID; //不再使用，一直为 0
    DWORD cntThreads; //该进程创建的进程数
    DWORD th32ParentProcessID;//父进程 ID
    LONG pcPriClassBase;// 该进程创建的进程的基本优先权
    DWORD dwFlags;// 不再使用，一直为 0
    TCHAR szExeFile[MAX_PATH];//指向该进程执行文件名字
//和路径的字符串的指针
} PROCESSENTRY32, *PPROCESSENTRY32;
```

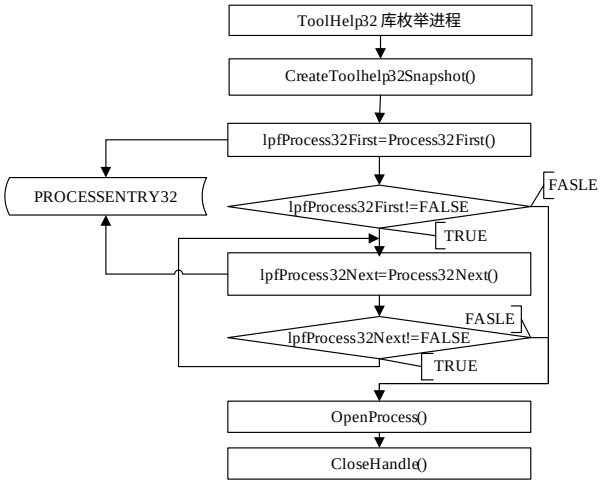

图 3 使用 ToolHelp32 库枚举进程的流程

在调用 Process32First() 前必须设置 dwSize 为 sizeof(PROCESSENTRY32)。通过把结构成员 th32ProcessID 传递给 OpenProcess() 就可以得到该进程的句柄，用 CreateToolhelp32Snapshot() 创建的句柄在使用后必须用 CloseHandle()关闭。

2.1.2 使用 PSAPI 库枚举进程

为了在 Windows NT 下获取进程快照，必须使用 PSAPI.lib 库中的 PSAPI 函数。PSAPI 库中与枚举进程有关的函数有：EnumProcesses()，EnumProcessModules()，GetModuleFileNameEx() 和 GetModuleBaseName()。EnumProcesses()函数的声明格式是

BOOL EnumProcesses(DWORD* pProcessIds, DWORD cb, DWORD* pBytesReturned);

参数 pProcessIds 的意义是：一个指向 DWORD 型进程数组 ProcessIds[]的指针，调用函数后该数组保存了当前的进程 ID 列表。

参数 cb 的意义是：以字节表示的传递给函数的进程数组的大小。

参数 pBytesReturned 的意义是：一个指向 DWORD 型数据的指针，调用函数后该该地址保存了已经使用的进程数组大小。

函数的返回值：成功返回非 0，不成功返回 0。

首先要调用 EnumProcesses()函数获取系统的每一个进程 ID。用表达式 pBytesReturned/sizeof(DWORD)可以计算出取得的进程数。由于无法事先知道进程的数量，也没有任何标志表明进程的数量是否超过进程数组的大小，为了获取全部进程的总数，如果返回的 pBytesReturned 等于 cb，就要采用更大的数组去重新读取进程，循环这个过程直到 pBytesReturned 小于 cb，这时进程数组 ProcessIds[]中才保存了系统中每个进程的 ID。

通过把 ProcessIds[]数组的元素 ProcessIds[i]传递给 OpenProcess()就可以得到该 ID 的进程句柄 hProcess。得到 hProcess 句柄后，调用 EnumProcessModules()则可以得到该进程的第 1 个模块句柄 hModule。EnumProcessModules()的声明格式是：

EnumProcessModules(hProcess, &hModule, sizeof(hModule), &pBytesReturned);

得到 hModule 句柄后用它作参数调用 GetModuleFileNameEx() 就可以获取进程的全路径名，或者调用 GetModuleBaseName()函数只可获得进程可执行模块名。这两个函数均带 4 个参数：进程句柄 pProcessId，模块句柄 hModule，返回路径名字的数组指针以及数组大小。用 EnumProcesses() 函数返回的每一个进程 ID 重复这个调用过程，便可以创建 Windows NT 的进程列表。整个流程如图 4 所示。

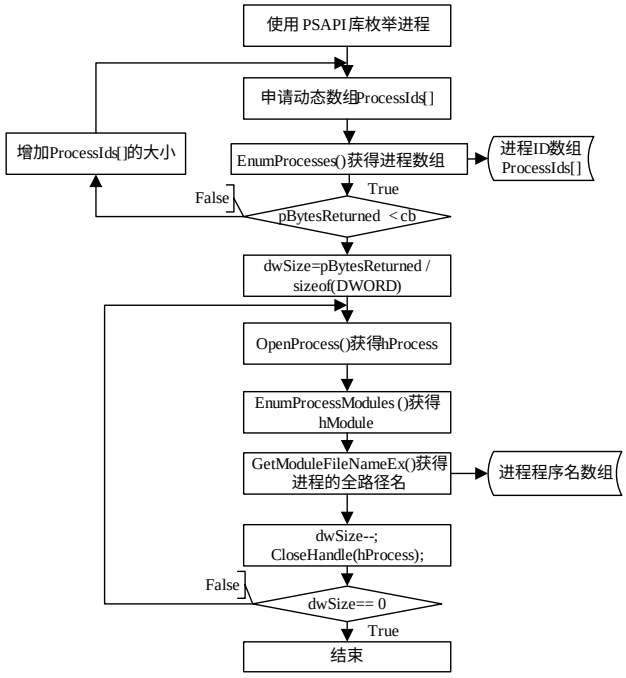

图 4 使用PSAPI.lib库枚举进程的流程

（下转第 259 页）