基于系统软件资源的应用软件功能集成策略

卜淮原　胡曙　（重庆后勤工程学院　630041）

摘要：本文围绕开发微机应用软件系统的工程组织问题，从操作系统的角度对应用软件的功能集成机理进行了分析，提出了“面向对象＋模块化＋系统软件融合调度”的软件工程组织策略。该策略强调系统分析以功能为划分对象的依据，并将其开发为可独立运行的模块，系统集成和功能调度以系统软件底层资源的基础，是一种可操作性和实用性较强，先进、可靠、易维护、易扩充的应用软件开发方法。

关键词：软件集成　软件工程　系统资源

1.概述

随着社会信息化的推进，微机应用在我国越来越普及，应用软件开发的需求越来越大，而开发力量却显得相对薄弱。许多单位尽管也拥有一定数量的软件技术人员，但开发、应用的水平仍然很低，许多本来可以由自己来开发的应用软件，却要花费巨额投资委托外部力量完成。造成这一现象的原因很多，但基本都是基于这样一个事实，即技术人员的软件设计、调试能力有余，软件工程和软件系统集成方面的知识不足。如何提高他们的软件集成能力，利用现有的软件资源，实现软件开发水平的质的飞跃，是提高我国计算机应用水平的关键，也是软件工程领域当前研究的热点之一。

软件工程尽管发展时间较短，但随着信息技术的飞速发展，不断有新理论、新技术涌现。“模块化方法”、“生命周期法”、“快速原形法”、“渐进获取法”、“面向对象(OOP)技术”、“第四代语言(4GL)”、“计算机辅助软件工程(CASE)技术”、“系统集成”、……，这些名词不但是技术发展的标志，也从一个侧面反映出“软件危机”依然存在。由于我国软件工程起步晚，上述技术、开发平台、运行环境(硬件和操作系统)大都是“引进产品”，文化、心理和思维习惯等方面与西方的差异，使得大部分“引进产品”在我国软件界生长缓慢。照抄照搬、亦步亦趋不利于我国软件业的发展。因此，适合我国国情的开发理论和技术方面的研究才显得尤为重要。

目前，“系统集成”在软件行业越叫越响，仁者见仁，智者见智，众说纷纭。就软件集成来说，实质无非是功能组合，即采众家软件功能之长为我所用。软件开发者大致是基于一定的开发平台，熟悉相关软件的接口要求，按照要求规范地开发接口软件部分和其他部分，然后用自己的界面实现总控。这种受到其他应用软件版权制约的集成策略，其软件工程的组织相对容易，本文不对其进行分析。操作系统是软件开发和运行的基本平台，本文要阐述的是仅基于操作系统环境，通过遵循某种规范或协议的积累开发，由操作系统的进程管理机制来完成组合的集成策略。

2.微机 Microsoft－DOS 操作系统中开发资源共享(可互用)软件机理分析

DOS是微机上使用最普遍的操作系统，基于它的应用软件数目众多。DOS中具备实时功能互用的软件大致有四种形式：

(1)设备驱动程序。这种程序由设备头、数据区、初始化过程、中断过程和命令处理程序五部分组成，用户必须严格按照 Microsoft 公司指定的规程进行编程。程序以内存映象文件的形式存在，其扩展名使用.BIN 或.SYS。装入系统是通过在系统配置文件 CONFIG.SYS 中的命令：

DEVICE＝驱动程序名.SYS(或.BIN)

来实现的，DOS 系统启动时，该驱动程序自动被装入内存。应用软件通过 DOS 的系统功能调用(INT 21h)实现对该驱动程序的访问和对相应设备的控制。此类驱动程序直接与硬设备打交道，屏蔽了机器的具体特性，方便了程序员的设计、编程，是基于设备共享考虑，实现虚设备”的一种技术。因此，它为软件的功能共享及其功能集成作了基础准备。此类程序尽管能实现共享功能的目的，但却有着明显的缺陷，一方面 Microsoft 公司关于此类程序的开发规范十分复杂、烦琐；另一方面此类程序是

在系统初启时一次性装入内存的，不管使用频度如何都要耗用较多的系统硬件资源。

(2)结束驻留程序(TSR 法 Terminate Stay Resident 或称"上托程序")。这种程序通过扩充 BIOS 中断功能，向用户提供更多或更强可供调用的中断服务程序。通过运行该程序，修改中断向量表，并将中断服务程序一次性装入内存的。应用程序通过调用 BIOS 软中断，或由用户按动"热键"来实现程序的"激活"。TSR 程序能处理低级信号，如特定的中断信号。这样处理中断的程序由于常驻内存，就为内存中各应用程序所共享。但此类程序除了也存在耗用较多宝贵内存资源的问题外，还因涉及大量直接硬件操作，给编程带来较多困难。

(3)重加载(具备 Shell 机制)程序。此类程序运行方式类似于多道程序环境之下的"父/子进程"运行方式。在此，本文借用进程概念来深入分析重加载程序的执行过程。"父进程"运行某个程序，在其运行当中，若需要运行另外某个程序，则由"父进程"生成某个"子进程"，且置"子进程"的代码段指针为欲运行程序的虚存(对内、外存进行统一管理的内存扩充技术)起始地址，然后将其投入运行。在"子进程"运行期间，"父进程"等待，待其执行完，撤销"子进程"，将控制权交回"父进程"，"父进程"从被中断点开始继续运行，从而实现了在程序一级的功能集成。

由于 DOS 并非是一个多用户、多任务环境，很多原本由系统来完成的工作(如生成"子进程"、置"子进程"的有关信息等)也不得不由"父进程"来完成，加重了编制程序的负担，而且由于 DOS 本身不提供内外存间交换(SWAP)功能，应用程序不欲加载的代码往往也会同时占据主存，使得主存利用率不高，对软件集成是极为不利的。

(4)并发程序。此类程序启动过程与"父子进程"类似，但控制方式不同，通过各自被分配的"时间片"，靠 DOS 系统时钟实现进程间切换。可实现"前后台"分别对应不同的处理。但进程间通信仍是十分困难的，也还存在耗用较多内存的问题。

上述四种程序的前三种执行状况都是顺序的而非并发的，控制取决于应用软件的逻辑结构。四种程序的通病是互用软件间的通信比较困难，要通过对用户并不透明的栈操作实现。软件集成涉及大量的底层操作约定(如寄存器操作、中断调用等)，对应用软件开发者来说是不易掌握的。由此可知，DOS 系统并发控制机制很弱，不具备现代操作系统的基本要求，软件集成受到很多制约，不是理想的开发平台。

3.微机 Microsoft Windows 系统并发进程调度机理分析

Microsoft Windows 是运行于 DOS 之上的"外壳"型软件，它以优秀的图形化界面、强大的功能和其上的众多工具软件赢得了绝大多数微机用户的青睐，成为微机应用软件开发和运行的主要平台。Microsoft Windows 针对 DOS 的不足，在许多方面进行了扩充，与 DOS 共同构成了一个具备现代操作系统特征的良好环境。就 Windows 的并发进程调度机制来说，大致有如下特点：

(1)Windows 是一种基于消息机制的多任务开发平台，支持"多任务"运行。多个窗口过程可同时处于 Windows 的统一管理下运行，靠各种各样的消息来决定系统的运行状态，使各窗口过程分别完成各自不同的功能。该性质与传统的多任务不同，传统的多任务是按硬件时钟分配给每一个程序一个时间片。Windows 仅在检查消息队列是否有消息时是多任务的。

(2)具有较完善的"消息循环"机制。Windows 采用"事件驱动"的方式，靠发送消息实现系统对窗口过程的调用(区别于 DOS 调用的"反调用")，窗口过程处理接收消息并执行相应的动作。"排队消息"和"非排队消息"的处理，使得 Windows 中各个不同进程得以协调运行。

(3)靠"句柄"访问对象，屏蔽了用户对物理设备的直接访问，由系统实施对各种临界资源的保护，对用户而言，操作系统变得"透明"，也更简洁。这与 DOS 环境下把全部资源交由用户控制的方式不同。

(4)用动态链接库(DLL. Dynamic Link Library)机制实现内存的动态覆盖，可做到"请求调入"。因此一个程序的代码长度允许超过内存能装入的最大容量。由于是动态链接，多个程序可以公用运行库中的子程序，大大减少了代码的冗余。该机制最大的优点在于尽可能提高主存利用率，给多进程的并发创造了良好条件。

(5)面向对象的进程管理。可方便地实现功能块代码与数据的"封装"、"继承"和"多态性"，利用"重载"特性可使得软件工程易于组织。

(6)良好的进程间通信机制。

①目标连接与嵌入(OLE)。该功能类似于 dos 的设备驱动程序中的中断处理机制,使用时无须"显式"调用,Windows 根据对象的数据类型自动予以对应的操作。

②动态数据交换(DDE)。该机制基于 Windows 的消息系统。两个 Windows 程序通过相互发送消息进行一次 DDE"会话"。这两个程序一个称为"服务器(Server),"一个称为"顾客(Client)"。DDE 服务器程序含有其他 Windows 程序有用的功能处理或数据。DDE 顾客程序从服务器获得功能服务或数据。若在程序中进行内存分配,则需使用 GMEM-DDESHARE 标志进行内存分配,使得该内存段可以在多个程序间共享。

③Windows 剪裁板。这是一种相对简单且开销较小的方法,程序可在剪裁板中存放数据,这些数据可以是文本、位图或图元文件,Windows 程序都可对其进行访问。剪裁板机制包含 Windows 中 USER 模块中的一系列函数,实际功能是实现程序间的内存块交换。

④在内存中共享在动态链接库中的信息。动态链接库可为多个 Windows 程序调用,它拥有自己的数据段、代码段等,可被多个进程所共享,也就是实现了在进程一级的资源共享和功能集成。动态链接库中的程序本身不能成为进程,没有自己的栈,而使用调用程序的栈。

以上分析表明,Windows 系统的进程管理能力很强,为实现并发任务间的功能共享提供了良好的环境。

4.基于并发进程的微机应用软件功能集成方法

软件工程涉及从系统调查分析、系统逻辑与物理设计、编程调试、测试验收、文档整理等各个过程,本文不对此作系统阐述,只描述系统功能设计阶段如何实现功能集成的方法。

优良的操作系统环境是系统集成的基础,具有较完善的进程管理功能是实现并发任务间功能共享的前提。软件功能集成的第一步应是选择合适的操作系统,如采用 Windows 这样的平台,则会有利于软件工程的组织实施。

操作系统选定后,第二步工作应是选定开发工具。如选用 Windows 中的"可视化(Visual 类)"开发工具,则需详细了解其在进程通信方面采用何种方式及其实现技术。

第三步工作是软件功能分解。要采用面向对象的方法把相近的功能归类,按不同的职能确定模块(Windows 中的独立程序或可为其他程序唤醒的命令),将其封装。各模块只需限定所完成的任务和功能程序类名,无须限定处理软件,让程序设计人员有较大的创造自由度。各独立模块按可单独运行的方式来编程、调试,目标代码尽可能采用动态链接库的技术进行连接。

第四步工作是设计"主界面"模块逻辑结构,该模块要利用操作系统的进程管理功能,在并发任务间实施调度,协调分配系统中所有功能模块程序完成约定担负的任务,最终进行"信息融合",实现系统全部功能。

第五步工作是界面设计,除规定所有模块的用户界面要遵循统一的风格外,还应约定不同模块的执行方式,前台还是后台运行。

第六步工作是进程间通信设计,在 Windows 中剪裁板和动态链接库方式的通信比较简单,动态数据交换则需要精心设计"握手协议"和数据格式。

最后,进行系统联调。

上述策略与传统做法的根本区别在于模块分解后的编程、调试工作是独立进行的,整个系统无须按处理逻辑循序开发,功能类调用由操作系统的并发进程管理机制完成,功能集成与系统融合成一体。该策略不仅实现过程简洁容易,而且充分利用了系统资源;不仅有利用积累开发,也给调试、维护带来极大方便。笔者用此方法成功地完成了 FoxPro for Windows 图象处理功能的扩充,实现了对图象的 JEPG 压缩、解压缩以及快速显示等功能。

5.结论

采用面向对象的系统分析设计方法,充分利用操作系统的进程管理机制来实现软件的功能集成,是一种科学可行的软件开发策略。

参考资料

[1] 张海藩　编著《软件工程》清华大学出版社

[2] Grady Booch.《Object Oriented Design With Application》Copyright 1991 By Benjamin/Cunmings Publishing Company, Inc.

[3] 夏东涛 编译《MS-DOS 高水平程序设计》电子工业出版社

[4] 吴之美等译《Microsoft Windows 3.1 程序员参考大全》清华大学出版社

(来稿时间:1996 年 6 月)