基于 Web 的 PDF 系统的设计与实现*)

Design and Implementation of PDF System based on Web

陈 明[1,3] 王彦丽[2] 刘 迅[1]

(石油大学计算机科学与技术系 北京102200)[1] (首都师范大学信息工程学院 北京100037)[2]

(合肥工业大学计算机与信息学院 合肥230009)

Abstract With the wide use of PDF documents, the common users' need for acquiring PDF documents conveniently is very urgent. This paper proposes a method of PDF on-line generating system. From the aspects of both theory and practice, this paper discusses the construction of the system and the optimization tactics in detail. The analysis and research on the load balance and the auto-dispatch is also made, and the result indicates that the algorithm and tactics are valid and practical.

Keywords PDF, Multithreading, Message queue

随着网络的日益普及,文档的通用性越来越重要。因此,对于文档的格式及跨平台性提出了更高的要求。

为了解决传输速度、文件格式和字体的限制、兼容不同的系统平台,Adobe 推出了 PDF(Portable Document Format)文件格式。这是一种通用文件格式,其跨平台、跨语言、跨软件的特性,可以使其运行于不同的操作系统和不同的程序语言版本中。它可以将文字、字型、格式、颜色及与设备和分辨率相独立的图形图像等封装在一个文件中,该格式文件还可以包含超文本链接、声音和动态影像等电子信息,支持特长文件,其集成度和安全可靠性都较高。PDF 文件使用了标准的压缩算法,易于传输与储存。它的页与页是相互独立的,可以单独处理各页,特别适合多处理器系统的工作。PDF 文件还包含文件中所使用的 PDF 格式版本,以及文件中一些重要结构的定位信息。由于 PDF 文件可以不依赖操作系统、语言和字体以及显示设备,就能逼真地将文件原貌展现给用户,因此越来越多的电子图书、产品说明、公司文告、网络资料、电子邮件开始使用 PDF 格式文件。PDF 格式文件目前已成为电子文档发行和数字化信息传播的一个标准。

面对 PDF 文档的广泛使用,有必要提供一个 PDF 文档在线生成的系统,从而可以使普通用户方便快捷地得到 PDF 文档。本文提出了一个在线 PDF 文档生成的方法。

1. 系统功能描述

系统的基本功能是:用户在客户端上载各种指定格式的文件,在服务器端通过 PDF 转换服务器进行转换,从而生成 PDF 文档;每个用户可以定制自己的转换配置文件和水印,进行加密设定和对自己的通讯录进行管理,生成后的 PDF 文件可以在线浏览或者通过邮件方式发送给指定用户;此外,还可以为管理员提供日志查询、用户管理、系统监控等功能。

2. 系统结构

系统采用的是 Browser/Server 的体系结构[1]。在逻辑上,主要是由以下几个部分组成:PDF 转换服务器、Web Service、历史记录管理 Service、任务队列。其结构与关系如图1所示。

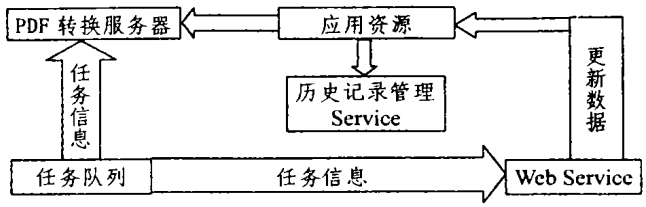

图1 系统结构图

2.1 PDF 转换服务器

PDF 转换服务器是系统的核心,主要用于监控系统的任务队列[2],按照某种优先级选择相应的任务节点,然后对任务进行转换处理,最终生成 PDF 文档。

首先,根据实际需要制定恰当的优先级原则。从任务队列中取出优先级最高的任务节点进行处理。由于这是一个多机多任务的环境,为了保证每个任务只被处理一次,采用了加锁保护机制。即当 PDF 转换服务器锁定一个任务节点后,立即对任务节点和与本任务有关的应用资源加只读锁,只允许其它进程查询任务或查询资源的状态,但不能进行其它任何操作。

PDF 转换服务器根据任务中待转换文件的文件类型,调用相应的应用程序,利用应用程序自身的打印功能来选择 PDF 打印驱动程序,对文档进行转换,最终生成 PDF 结果文档。在进行转换的过程中,PDF 转换服务器还要根据进展情况来记录或修改数据库中该任务的状态信息,并保存到日志文件中。

PDF 转换处理程序在处理每一个任务的同时,都要启动相应的记时线程,以记录本次转换所用的时间。在判断任务超时的时候,将会用到这个记时线程。

2.2 Web Service

Web Service 运行在 Web 服务器端,它的主要任务是发送消息给系统,通知系统有新任务产生;监控任务运行状态,更改数据库中任务的相应信息。其工作过程如下:(1)根据客户请求,向系统消息队列发出特定消息,表明有新任务产生;

*)国家自然科学基金资助项目(60072006,19971058)、石油大学(北京)校科研基金(99-I-01)项目。陈 明 教授,博士生导师,主要研究领域为计算智能,分布式并行计算。王彦丽 讲师,主要研究领域为计算机网络。刘 迅 硕士,主要研究领域为计算智能,分布式并行计算。

(2)该 Service 监控任务队列,对任务队列中三种状态的任务进行跟踪,这三种状态分别是运行、失败和完成。如果队列中当前的任务是处于运行状态,则该 Service 根据该任务的标识符获得它的记时线程的句柄,从而可以与记时线程进行通信,获得该任务的运行时间。如果运行时间超过限定值,表明本次转换超时,系统需要对任务节点进行解锁,释放资源,关闭记时进程,并且将任务状态置为"失败";否则,将本次任务插入到任务队列的尾部。如果队列中当前的任务是处于完成状态,则对任务节点进行解锁,释放资源,关闭记时进程,将任务从队列中删除,并将数据库中的任务状态置为"成功"。如果队列中当前的任务是处于失败状态,则将任务从队列中删除,将数据库中的任务状态置为"失败"。

2.3 历史记录管理 Service

对于一个完善的应用系统来说,历史记录的管理是必不可少的。它不仅可以提供整个系统的运转记录,反映整个系统的运转情况;更重要的是,能够对所保存的数据进行分析、统计和规划,从而发现系统潜在的问题,为日后系统的维护和完善打下基础。

历史记录管理 Service 对所有用户已完成的任务进行统一管理,包括分类、排序、删除和写日志、备份等。实际上,该 Service 对数据库中已完成的任务记录进行全面分析和比较,其中包括任务的类型、时间、完成情况等。根据系统的实际需要,将这些任务的具体数据进行统计、分类和排序,并且根据统计数据作出数学模型,将其写入历史日志中以供管理员查询。同时,该 Service 能够提供备份服务,它可以根据管理员的要求将各种日志文件定期存档。

2.4 任务队列

任务队列是构建的一个数据结构,它记录的是系统中待处理或正在处理的任务信息。任务队列中的每一个节点主要由任务标识符、任务类型、任务状态、访问标志位等组成。任务队列在系统初始化时被激活。

在正常运转的系统中,这四个部分都不能单独工作,它们之间需要进行相互通信,传输数据,这样才能准确协调地工作。

3. 系统实现

这个系统是建立在 Browser/Server 体系结构上的,涉及到操作系统的进程控制和多线程操作、文件系统、数据库[3]等方面。这里将着重阐述系统的核心问题:各 Service 间的协同工作[4]、多台 PDF 转换服务器的自动调度问题。

3.1 Service 间的协同工作

系统的关键部分在于 PDF 转换服务器与 Web Service 之间是否能够协同工作。为了达到这一点,就必须保证这两个 Service 的进程能够进行正确合理且高效的通信。

在最初的方案中,这两个 Service 并没有直接地进行通信,而是通过查询或更改任务队列中任务的状态和数据库信息,间接地交换数据进行通信。通过一段时间的测试,发现这种方案在数据量较小和用户较少的情况下,性能尚可。一旦遇到大数据量、多任务的情况,系统便出现阻塞现象,系统性能急剧下降,大量任务堆积,转换进展缓慢。为了解决这个问题[5],对方案进行了改进,放弃了原有的间接通信机制,采用直接通信。也就是说,利用消息机制在 PDF 转换服务器与 Web Service 之间传递信号,进行通信。

具体如下:

1. 利用操作系统提供的函数,在系统中注册一个新的消息。

2. 当出现新任务时,Web Service 向系统消息队列发出这个消息,通知 PDF 转换服务器有新任务到达,请求处理。

3. PDF 转换服务器可以对消息进行过滤和探测,专门捕获这个新注册的消息。当转换服务器获得这个消息时,便可根据消息的参数得到该任务的信息,判断是否可以得到该任务的控制权。如果可以得到控制权,就对任务节点和应用资源加锁,然后 PDF 转换服务器就可以进行转换了。否则,放弃这个消息,等待下一条消息的到达。

实践表明,这种经改进的方案无论是在时间上、效率上,还是系统资源的使用率上,都有着较大的提高。

3.2 多台 PDF 转换服务器的自动调度

考虑到系统的性能和负载均衡[6],在本系统的解决方案中支持多台物理的 PDF 转换服务器。这样,在每台 PDF 转换服务器上均安装相关应用软件及 PDF 转换服务程序,这些转换服务器共同分担系统的任务[7]。然而这时,多台 PDF 转换服务器的统一调度便成为首要问题,因为调度的不当可能会使多台转换服务器处理同一个消息,造成系统性能的下降和系统资源的浪费。

假设:系统中有 M 个 PDF 转换服务器,其中 P 个空闲,有 N 个任务同时到达。当 P>=N 时,不做讨论。当 P<N 时,只有 P 个任务被处理,剩余(N-P)个任务将等待处理。这(N-P)个消息将被传递到 M 个转换服务器上,即转换服务器将存储(N-P)M 个消息。由于一个任务只可能被处理一次,因此(N-P)M-1个消息实际上是无效的,系统对它们的处理将是徒劳的,只会占用系统资源。

因此,在服务器端嵌入自动调度模块,就可以根据实际情况对消息队列进行控制,有效地针对某个或某些 PDF 转换服务器发送消息,避免了系统对无效消息的处理。这样,就可以实现多台 PDF 转换服务器的高效调度,均衡了系统的负载,提高 PDF 的转换速度。

3.3 自动调度算法

自动调度算法[8]采用了类似 Oracle DLM(分布式封锁管理)封锁机制[9]的原理,使用了两个消息队列:系统消息队列和任务消息队列。在系统消息队列中有新任务建立时产生的消息,任务消息队列存放的是发送到 PDF 转换服务器上的消息。实际上,任务消息队列是一个具有存储转发功能的数据结构。系统消息队列中相应的消息并不是直接发送给 PDF 转换服务器,而是先要将这个消息插入到任务消息队列中,再由任务消息队列进行转发。PDF 转换服务器是从任务消息队列中取出消息,进行处理。这样再通过一些控制操作,就可以实现消息的有效传递与处理[10,11],避免某些消息被重复处理,参阅图2。

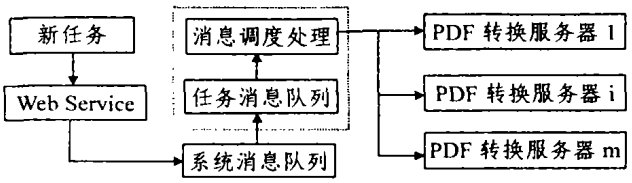

图2 自动调度算法

调度算法描述如下:

//调度进程

```
do while True
wait_for_sys_msg(msg)//等待系统消息队列中的消息
insert_msg_queue(msg)//将消息插入任务消息队列中
if 不是暂停状态 then
        forward_to_everyone(msg,status) //转发任务队列的消息
        if status=True then //消息被空闲的转换器接受
          update_queue()//删除该消息,更新任务消息队列
        else //所有的转换服务器都忙
             update_status(暂停) //暂停发送消息,转入暂停状态,
             wait_for_wakeup(msg) //等待空闲转换器的应答
        end if
end if
loop
//PDF 转换服务进程
do while True
wait_for_task_msg(msg) //等待任务消息队列中的消息
status=isWorking()//本台转换器是否在工作
if ( status==False) then //空闲
        getTask(msg,task)//根据消息得到任务节点的信息
        if ((task 存在) and (task.locked==False)) then //任务节点
没有被锁定
             lock_task_node()
             lock_app_resource()
             answer_msg(True)//向调度进程发出应答信号 true
             execute_job()//调用转换模块(异步执行)
             //完成之后发出唤醒信号 给自己和调度进程
        update_task_queues()
end if
else //机器忙
     answer_msg(False)//向调度进程发出应答信号 false
     sleep_wait_for_wakeup()//进入休眠,等待唤醒
end if
loop
```

4. 性能比较

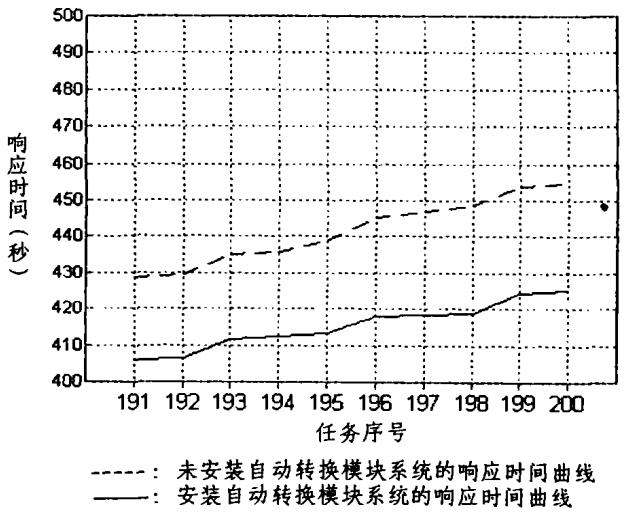

图3 统计图

为了验证自动调度模块的功效,测试其对系统性能的影响,对安装自动调度模块前后的两套系统进行了对比。进行对比测试的参数主要是系统响应时间和文档生成时间。测试环境为:1台 Web 服务器,3台 PDF 转换服务器,5个客户端,15秒内向系统提交200条任务,每个任务均为400K 的 Word 文档。表1为最后10条记录的统计数据。

表1 统计数据

时间 / 任务	未安装自动转换模块的系统		安装自动转换模块的系统	
	响应时间(秒)	生成时间(秒)	响应时间(秒)	生成时间(秒)
191	428.6	6.4	405.9	6.2
192	429.3	6.2	406.4	6.1
193	434.9	6.4	411.7	6.3
194	436.1	6.3	412.2	6.2
195	438.7	6.3	413.1	6.3
196	445.1	6.2	417.8	6.2
197	446.9	6.1	418.4	6.2
198	448.7	6.3	418.6	6.4
199	453.9	6.4	424.2	6.3
200	454.7	6.3	425.4	6.3
合计(秒)	4416.9	62.9	4153.7	62.5

图3是统计结果,从实验结果可以看出,本文提出的自动调度算法能够更充分利用系统资源,提高并行效率,而且在多用户环境下,效果更加明显。

结论 本文提出了一个 PDF 在线生成系统的方案。这项方案采用了基于 Web 的计算模式,充分利用操作系统提供的各种功能,优化了 PDF 的转换过程,在实际中取得了不错的效果,具有很强的实用价值。

参 考 文 献

1 Gellersen H-W, Gaedke M. Object-Oriented Web Application Development. IEEE Internet Computing, January-February 1999
2 Tanenbaum A S. Modern Operating System. Englewood Cliffs, NJ: Prentice-Hall, 1992
3 Corrigan P, Gurry M. Oracle Performance Tuning
4 Arlitt M F, Williamson C L. Web server workload characterization, 1996
5 Jain R. The Art of Computer Systems Performance Analysis. New York: John Wiley, 1991
6 Macehiter N. Web server performance and scalability
7 Anderson B, Gossain S. An Iterative Design Model for Reusable Object-Oriented Software. In: Proc. OOPSLA'90
8 LIU T D C, Layland J W. Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. Journal of the ACM, 1973
9 Greene J. Oracle DBA Survival Guide
10 Hardy, James K. Inside Networks. Upper Saddle River, NJ: Prentice-Hall, 1995
11 Benage D, Mirza A. Using Visual Studio, Que Corporation