

基于多应用服务模式的视频转换中间件的开发和实现

张祖礼, 何洪波, 尹海龙

(中国科学院 计算机网络信息中心, 北京 100190)

摘 要: 本文从中国科学院网络化科学传播平台视频资源展示的实际需求出发, 提出了一种基于多应用服务模式的视频转换中间件的解决方案. 重点介绍了“视频转换中间件”的技术架构和系统流程, 并详细介绍了其功能模块的设计和主要算法. 通过“视频转换中间件”的实施, 可以有效提高系统的可重用性, 有效实现网站群视频资源的整合, 提高网站群对外服务的效能.

关键词: 视频转换; 中间件; 多应用服务模式; 视频资源整合

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2013)02-0042-04

The Development of Video Conversion Middleware Based on Multi—Application Service Mode

ZHENG Zu-li, HE Hong-bo, YIN Hai-long

(Computer Network Information Center of CAS, Beijing 100190, China)

Abstract: Starting from the showcase demand of video resources in the Internet—based Communication Platform of the Chinese Academy Sciences, this paper proposes a solution of a video conversion middleware based on multi—application service model. The authors develop a focused introduction to technology architecture and system processes of "video conversion middleware", and a detailed description of the design and the main algorithms of the system functional modules. This paper argues that the implementation of the "video conversion middleware" can effectively improve the reusability of the system, and achieve the integration of video resources in the site group in an efficacious way. It improves the performance of the external services of the website group.

Key words: video conversion; middleware; multi—application service model; integration of video resources

1 引言

科普视频资源是很重要的一类科普资源. 为了能够以统一方式向公众提供视频服务, 满足平台的集成和统一展示要求, 通常需要将原始的各类形态的资源进行统一的转换. 但是对于一般中小型的科普网站来说, 视频资源只是众多类型资源的一部分, 视频科普的形式也不是其主要形态, 开发一个完整的视频转换功能或模块并不现实. 因此, 本文提出基于多应用服务模式, 提供视频转换功能的中间件平台的实现方案, 为多个需要视频转换功能的应用提供个性化的配置方案和统一的调用接口. 特定的

应用通过调用视频转换中间件, 即可完成对多种格式的视频文件的统一格式转换^[1-2].

2 系统结构设计

视频转换中间件将作为操作系统的一个服务进程运行, 包括视频转换、调度控制和安全控制三个部分. 其系统结构如图 1 所示.

经过验证的外部应用程序可以通过标准的 HTTP 或者 FTP 文件上传方式, 将原始的视频文件上传到视频转换中间件视频存储区的指定目录中. 中间件定期扫描储存区, 并自动获取视频基本信息. 中间件根据上传视频所属应用的配置信息, 采用

收稿日期: 2012-05-18; 修回日期: 2012-06-12

基金项目: 中科院计算机网络信息中心青年基金

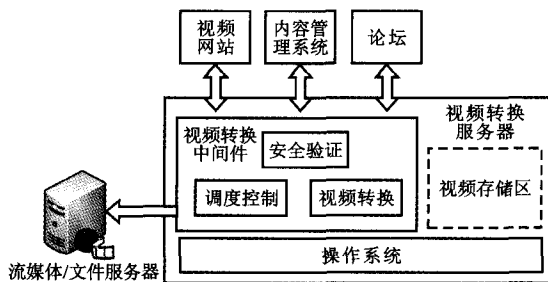


图1 系统结构图

合理的调度策略调用视频转换模块对该视频文件进行格式转换,并将转换进度以消息的方式通知外部应用程序。转换完成后,视频文件将转移到该应用指定的目标地址或可访问的服务地址,供应用的后续使用。在整个与中间件交互的过程中,视频转换服务的运行是在后台自动进行的,正常情况下不需要人工干预。在视频转换的各个关键环节,中间件通过接口向外部应用发送视频转换进度和状态等消息,外部应用也可通过接口向中间件发送控制请求,实现可控的文件转换处理。

3 功能模块设计

视频转换中间件系统包括视频文件传输模块、视频文件信息预处理模块、文件格式转换调度模块、视频文件格式转换插件接口模块、应用配置管理模块和应用交互接口模块。视频文件传输模块是一套用于在应用与转换服务之间进行视频文件传输的处理功能,主要采用 HTTP、FTP 等标准的文件传输协议;视频文件信息预处理模块主要负责从视频文件中提取视频信息和音频信息,并初始化视频转换环境;文件格式转换调度模块采用应用权值累加的调度算法在多个转换任务中选择最合适的任务进行格式转换;视频文件格式转换插件接口模块是提供将第三方的视频格式转换程序嵌入到中间件系统后的操作接口,在中间件内部调用该模块提供的相关操作接口完成相应的转换任务,不需要关心具体的转换算法等的实现细节;应用配置管理模块是每一个应用加入到中间件平台所需要的参数信息的配置和管理模块,负责解析配置文件的参数,生成转换所必需的执行命令等;应用交互接口模块是中间件与应用程序进行交互—反馈的通讯模块。系统流程图如图2所示。

3.1 文件传输模块

文件传输模块在应用程序与视频格式转换中间件之间建立起视频文件双向传输通道;应用程序将

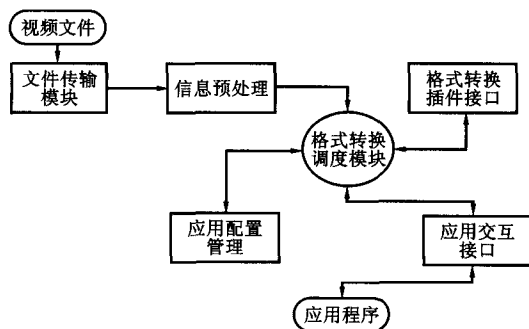


图2 系统流程图

原始的视频文件传输到中间件系统,中间件系统将转换完成的视频文件发送给应用程序。文件传输的方式可以采用 HTTP 和 FTP 等通用的文件传输协议,传输模块提供统一的调用接口,具体的传输方式的实现细节可以通过中间件的配置文档进行设定,应用程序具体采用的传输方式将由应用配置文档进行设定^[3]。

文件传输模块的结构如图3所示。

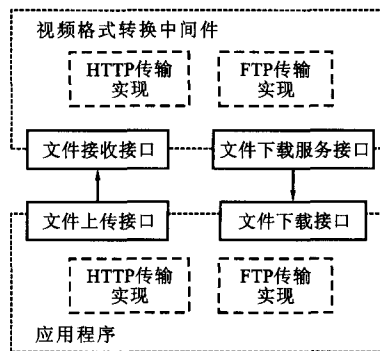


图3 文件传输模块结构示意图

文件传输模块对外提供的接口按功能可以分为文件接收接口、文件上传接口、文件下载服务接口和文件下载接口。这些接口分别提供给中间件和应用程序调用,HTTP/FTP 传输实现类将完成具体的文件传输功能。

3.2 文件信息预处理模块

文件信息预处理模块主要负责获取上传到中间件的视频文件的格式信息,并根据视频文件的清晰度和视频文件的播放时间长度,结合系统预设的权值参数,生成视频文件初始的权值信息。

视频文件信息的获取是通过调用视频文件格式转换插件接口模块提供的相应接口实现。

权值信息将会作为文件格式转换调度模块进行转换调度控制的起始依据。视频文件生成初始的权值的计算公式如下:

$$WI = Initpriority - \frac{dur}{df} - \frac{rs}{rf} \quad (1)$$

式中, WI 是视频文件初始的权值, InitPriority 是系统设定的系统视频文件初始权值, dur 是视频的播放时间长度, rs 是视频文件的清晰度, df 是视频的播放时间长度的比例因子, rf 是视频文件的清晰度的比例因子. 式(1)所示, 视频文件的转换起始权值是由视频的播放时间长度和视频文件的清晰度决定, 其中视频的播放时间长度的公式如下:

$$dur = \begin{cases} dur, & dur < MD \\ MD, & dur > MD \end{cases} \quad (2)$$

式中, MD 为最大视频播放时间, 这个参数是系统预设的视频播放时间最大值, 设置这个值是为了减少播放时间过长的文件在转换过程中的等待时间. 视频文件的清晰度的公式如下:

$$rs = \begin{cases} rs, & MinR < rs < MaxR \\ MinR, & rs \leq MinR \\ MaxR, & rs \geq MaxR \end{cases} \quad (3)$$

式中, MinR 是最小视频分辨率, MaxR 是最大视频分辨率. 这个公式用来规范转换视频的分辨率区间, 从而保证分辨率在权值中的影响在一个指定的范围内.

3.3 文件格式转换调度模块

文件信息预处理完成后, 视频文件被加入到文件格式转换调度队列中, 以后的处理将由文件格式转换调度模块进行控制. 具体的流程如图 4 所示.

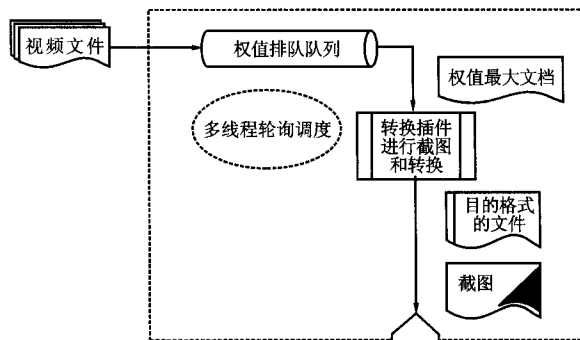


图 4 文件格式转换调度流程

视频文件格式转换调度流程描述:

- (1) 视频文件根据初始权值的大小进行排队, 权值最大的文件排在队列的最前端;
- (2) 视频转换最多并发数, 由应用配置管理模块中的中间件配置部分设定;
- (3) 转换过程没有抢占调度方案, 一旦一个文件开始转换, 只有用户可以通过应用交互接口模块将转换进程取消;
- (4) 每隔一定时间, 排队中的所有视频的权值就相应增加, 增加的权值由应用配置管理模块设定, 以

保证在有视频完成转换后, 排队中的初始权值较小的视频不会饿死;

(5) 使用视频转换插件对权值最大的视频进行转换, 获得文件的截图和目的格式的视频.

在视频转换任务在队列中排队过程中, 视频文件的权值变化如下:

$$PW = \begin{cases} WI, & t=0 \\ WI+t \times aw, & t \geq 1 \text{ 且 } PW < \text{MaxPW} \\ \text{MaxPW}, & PW \geq \text{MaxPW} \end{cases} \quad (4)$$

式中, PW 是视频文件在队列中的权值, t 是转换调度轮询的次数, aw 是应用的加权值, MaxPW 是系统设定的最大队列权值.

针对每一个应用, 应用配置管理模块会为其设定一个系统的参数, 这其中就包括应用的任务优先级信息, 这个信息就标识了每隔一定时间后增加权值的多少. 应用的任务优先级越高, 应用的加权值就越大, 应用的视频文件在队列中的等待时间就越短, 从而在应用级别上进一步地保证了高优先级视频的及时地转换调度^[4-5].

3.4 格式转换插件接口模块

本方案中, 视频格式转换中间件没有直接提供视频格式的转换功能, 而是使用格式转换插件接口模块将提供视频格式转换功能的相关应用程序以统一的调用接口方式封装起来. 用户可以选择的方案很多, 对于商用的、开源的和用户自行开发的视频格式转换应用都可以简单地与中间件进行集成.

处理类只需要继承 BaseHandler 抽象类, 需要实现 prepare 和 execute 两个接口, 所需要处理的信息将由 message 属性给出. BaseHandler 的类结构如图 5 所示.

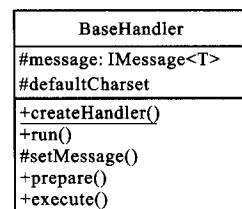


图 5 BaseHandler 的类结构

对于视频格式转换插件, message 实现类是 ProcessMessage<Task> 类, 这个类中存储了任务的实体对象, 进而包含了视频格式转换过程中所需的所有转换相关的基本信息. 在调度模块开始调度转换进程时, 视频格式转换插件的实现类就可以根据任务的状态和视频文件信息对视频文件进行格式

转换。

3.5 应用配置管理模块

应用配置管理模块包括两个方面,一是中间件平台的相关配置;二是应用程序客户端的配置信息。中间件平台的配置主要是系统监听的端口、进程和线程池的大小、调度的周期长度、默认的通讯实现类、默认的调度实现类和默认的格式转换插件类等,这些配置与中间件的基本运行环境相关。

应用程序客户端的配置信息中包括应用的基本信息、通信所用的字符编码、任务类型、截图参数、转换参数和通信时延等。在中间件启动过程中,首先加载中间件平台的配置信息,然后依次加载多个应用程序客户端配置信息。加载应用程序客户端配置信息的过程中,中间件在服务内部创建多个与应用程序对应的代表类。在以后通信和转换的过程中,中间件调用应用代表类提供的相关方法,最终实现与应用的通信和交互。

3.6 应用交互接口模块

中间件与应用程序的交互方式如图6所示。使用了传统的TCP网络通信协议,服务中间件与每一个应用程序建立一条TCP连接。应用交互接口模块就是为中间件与每一应用进行通信提供必要的交互接口,这一层的封装方便了用户重写通讯实现类、调度实现类和格式转换插件类,使这些重写的实现类的功能更好地与应用程序进行信息交互。

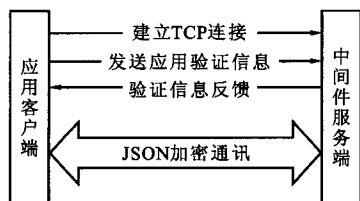


图6 中间件与应用程序的交互

在完成与应用建立通信连接之后,应用程序会向中间件发送应用验证请求,验证请求中包含应用的公钥数据和应用的基本信息,中间件在验证后,向应用程序发送反馈信息。在这之后,应用程序与中间件之间使用JSON字符串的方式将通信信息类序列化后,传输到另一方,在这个过程中使用应用程序配置的通信密钥对通信数据进行加密,并附着在通信数据之后一起传输。下面是原始的通信数据JSON格式的示例:

```

{"type": "2", "tid": "1", "tresult": "1", "video": { "type": "2", ... }, "audio": { "type": "2", ... }
  
```

在数据到达接收方后,接收方使用通信密钥对

原始数据再次加密,用得到的数据与传入的通信数据加密信息进行比较。如果加密信息一致,接收方就开始处理和解析请求。

4 结束语

基于多应用服务的模式在很大程度上提高了系统的可重用性,一次开发或购买的视频格式转换应用程序可以为很多个应用程序的开发服务,降低了其他应用程序在开发过程中的技术要求。视频转换中间件在解决对视频转换功能有一定需求的网站(如内容管理系统、论坛和中小型的视频网站)开发方面,能够起到缩短开发周期、降低系统运行风险的作用;在大规模的网站整合的过程中,有关视频方面的转换和播放服务可以借用本方案进行合理地资源统一规划和整合。

本方案所涉及的视频转换中间件是单服务器模式,由于服务器性能方面的制约,在并发进行大规模视频转换服务时转换调度达不到高效的需求,队列中的大视频的等待时间很可能达到系统设定的最大队列权值(MaxPW)。在未来的开发过程中,可以考虑使用多服务器的集群方式,加大转换过程中系统的并行运算效率,从而为更大规模的视频转换提供服务。

参考文献:

- [1] 王虎. 基于中间件技术构建数字视频网络应用平台[J]. 广播与电视技术, 2007(12): 44-49.
- [2] 李璇. 多媒体展示框架中间件的设计与实现[D]. 上海: 复旦大学, 2011.
- [3] 陈子良. 数字流媒体中间件研究与实现[D]. 武汉: 华中科技大学, 2009.
- [4] 段艳明, 肖辉辉. 基于Java NIO开发高性能并发型服务器程序的研究[J]. 软件导刊, 2009, 8(5): 63-66.
- [5] 陈兆荣. 流媒体数据分发技术研究与实现[D]. 广州: 华南理工大学, 2010.

作者简介:



张祖礼 男, (1985—), 硕士研究生, 工程师。研究方向为基于web的流媒体技术、数据网格中间件。

(下转第49页)

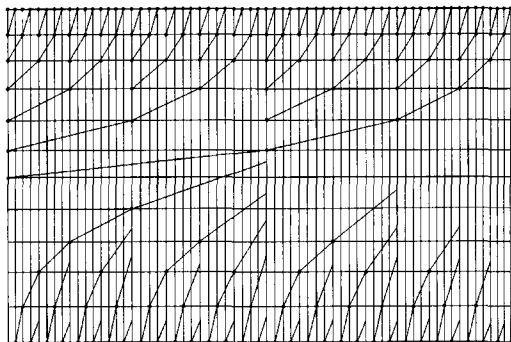


图5 64位 Brent Kung 树形结构

6 设计结果

本文设计的 32 位有符号/无符号乘法器采用 Verilog 语言编写代码,并通过 Modelsim6.5 仿真验证,仿真结果如图 6 所示。

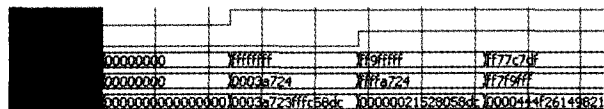


图6 Modelsim6.5 仿真结果

其中 rst 为复位控制信号, rst 为 0 时执行复位操作, 为 1 时执行正常的计算操作. Sign 为数据类型标识信号, sign 为 0 时执行无符号乘法操作, 为 1 时执行有符号数乘法操作. 测试数据的选取采用的是全面性和随机性相结合的方法. 乘数 Y 取 3a724、fffa724 可遍历改进的 Booth 编码产生的 8 种输入情况产生的 5 种输出结果. 随机数选取可以检查 Wallace 树、Brent Kung 树的正确性. 通过仿真结果可知, 本设计可对 32 位有符号数、无符号数正确计算得到 64 位结果, 满足设计要求。

设计采用 Xilinx 公司的 XC2VP100 为目标芯片, 综合结果的延时为 39.1ns, 使用 1485 个 slice、2397 个 LUT 及 110 个 IOB. 以 SMIC 的 0.18 微米标准库为目标库, 使用 Synopsys 的 Design Compiler 进行综合得到本设计的最小延时为 4.2ns, 所需

面积为 0.1629mm². 其中采用 7:2 压缩的 wallace 模块延时 1.9ns, 占用面积 0.0674mm²; 64 位 Brent Kung 加法器模块延时 1.66ns, 占用面积 0.007mm². 与文献[5]的由 3:2 和 4:2 压缩器构成的新型 Wallace 结构及设计结果相比, 速度提高了 3%。

7 结束语

文中介绍了一种有符号/无符号的 32×32 乘法器的实现方法, 使用一个符号标识位识别有符号数和无符号数操作, 通过改进的 Booth 编码减少了部分积个数, 并通过符号延伸的优化减少了中间资源消耗, 对部分积进行统一的符号操作, 简化了程序设计的复杂性. 本设计采用了 7:2 压缩结构的 Wallace 树及 64 位 Brent Kung 树结构超前进位加法器, 有效地提高了计算速度. 仿真验证了设计的正确性, 并通过综合得到性能参数, 整个设计简单高效。

参考文献:

- [1] 吴辉, 张磊. 一种新颖的可重组乘法器设计[J]. 微电子学与计算机, 2004, 22(9): 161-163.
- [2] 刘东. 采用 Booth 算法的 16×16 并行乘法器设计[J]. 现代电子技术, 2003(9): 21-22.
- [3] 汤晓慧, 杨军. 基于 Booth 算法的 32×32 乘法器的 IP 核设计[J]. 电子器件, 2005, 28(1): 218-220.
- [4] 陶智德, 林涛. 32 位高速复合加法器设计[J]. 电子工程师, 2004, 30(11): 22-23.
- [5] 李军强. 32×32 高速乘法器的设计与实现[J]. 微电子学与计算机, 2009, 26(12): 23-26.

作者简介:

朱鑫标 男, (1988—), 硕士研究生. 研究方向为数字集成电路设计.

施隆照 男, (1968—), 副教授, 硕士生导师. 研究方向为集成电路芯片设计.

(上接第 45 页)



何洪波 男, (1970—), 硕士研究生, 高级工程师. 研究方向为大规模科学数据管理及应用、基于 WEB 的数据可视化处理.



尹海龙 男, (1983—), 工程师. 研究方向为多媒体与交互技术.