

# 多媒体协作平台中录制系统的研究与实现

姚毅, 单宝松

(北京航空航天大学 软件开发环境国家重点实验室, 北京 100083)

**摘要:**在多媒体协作平台中,录制系统将实时协作过程中的音视频数据存储下来,用户通过回放可以重温整个会话过程。文中针对录制系统数据量大、实时性高、并发访问多的特点,研究并实现了一个实时多媒体录制系统。就缓冲管理、存储策略、会话管理等关键问题进行了分析并给出了解决方案。另外,为了提高系统各节点利用率和自适应性,提出了基于预测和负载反馈的动态负载均衡机制以及录制任务迁移策略。

**关键词:**协作平台;录制系统;数据存储与组织;动态负载均衡

**中图分类号:** TN919.8

**文献标识码:** A

**文章编号:** 1673-629X(2007)09-0163-04

## Design and Implementation of Record System in Multimedia Collaboration Platform

YAO Yi, SHAN Bao-song

(National Key Lab. of Software Development Environment of Beihang University, Beijing 100083, China)

**Abstract:** In the multimedia collaboration platform, the record system stores the real video and audio data of the collaboration process, thus users can playback and review the whole recorded session later. This paper analyzed the difficulties of the record techniques: vast number of data, concurrence, real-time requirement, and proposed and implemented a desktop environment-faced record system based on RTP protocol. Furthermore, the paper discussed some key issues in the implementation of record system: buffer management, storage policy, session management, etc and presented related solutions meanwhile. Besides, to improve the scalability and maximize the utility of nodes in the system, proposed a dynamic load balancing mechanism based on both load information and storage prediction and record task immigration scheme.

**Key words:** collaboration platform; record system; data storage and arrangement; dynamic load balancing mechanism

## 0 引言

Internet 以及多媒体技术的巨大发展促进了基于多媒体的协作环境的发展,如视频会议系统、远程教育、远程医疗等,它们将时间上分离、空间上分布而工作上又相互依赖的各个协作成员通过文本、图片以及更重要的音视频实时交互方式,有机地组织起来,以完成某一任务。这些协作平台实现了资源的共享以及远程的协助,改变了人们原有的工作和科研方式,大大提高了工作效率。其中一个重要的问题就是,如何将协作过程中各节点多媒体数据实时录制下来,这样错过现场或不方便与会的用户可以通过回放而重温整个会话协作过程,或者录制的音视频也可作为历史资料存档以备以后参考。

相比文本、静态图片等离散媒体的存储,多媒体协作平台中的录制系统<sup>[1]</sup>面临更多挑战:首先,对于如同视频会议系统等协作环境,往往多个会话同时进行,而每个会话又有来自不同节点的数十路左右的音视频流,而相对应地,录制系统就是要把这些实时数据全部或部分存储下来。一方面,存储的多媒体数据量巨大,存储压力大;另一方面,要求录制系统能够同时录制较多的并发流,系统吞吐率要求较高。然后,实时性要求高,要存储的数据来自网络上的实时报文,所以系统需要合理的数据组织策略来最大化 I/O 访问效率,防止 I/O 过于忙碌而使空间“溢出”,丢失数据;另外,由于音视频的连续性特点,同一个会话的音视频流间以及来自不同节点的视频流间都存在一定的时序同步关系,录制系统应有相应机制以便于回放时的媒体集成与同步,进而还原会话的完整历史场景。最后,各个会话录制任务间应相互独立,即系统需要会话管理。

笔者以桌面视频会议系统为典型场景,基于普通 PC 环境,研究并实现了一个分布的录制系统。

收稿日期:2006-11-27

基金项目:国家科技基础条件平台项目(2004PT101067)

作者简介:姚毅(1982-),女,山西介休人,硕士研究生,研究方向为多媒体技术与应用。

## 1 系统体系结构

### 1.1 系统总体结构

在本系统中,要录制的实时数据来自 Mbone 应用工具集衍生出的 Admire 视频会议系统<sup>[2]</sup>,其数据传输基于 RTP/RTCP 协议。其中,RTP 报文承载会话中的音视频数据,而通过 RTCP 报文来监测会议进行中节点加入或退出的变化状态,并获取每个节点描述信息以及接收的媒体质量。另外,如同大多的视频会议系统,Admire 系统是基于组播通信的,并且一个逻辑会话的音视频会唔分别对应一个组播地址。因而,文中设计的录制系统也是基于 RTP 协议的,其由媒体服务器集群、监听器、客户端三部分组成,如图 1 所示。

(1)媒体服务器集群由  $n$  个分布的存储资源(每个节点有多块硬盘存储器)和计算能力等性能良好的录制服务器节点组成,并由一个中央控制器对录制任

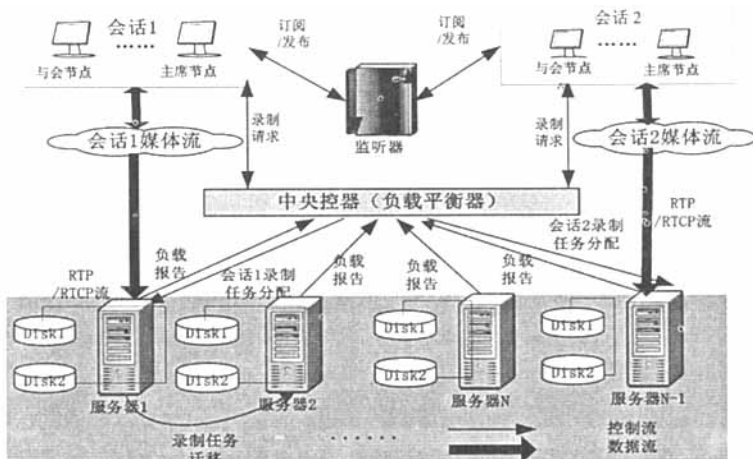


图 1 系统体系结构图

务进行全局统筹调度以保证系统负载基本均衡,使系统性能最大化。该集中式的调度控制简单,且由于中央控制器只执行控制命令,计算与通讯量不是很大,发生单点瓶颈的概率较小。当系统规模增大时,也可为中央控制器增加若干后备节点,缓解单点压力。中央控制器的具体工作是维护各节点负载、录制会话等状态信息,当接收客户录制请求后,选择一个较优的调度方案,把新的录制任务分配到目标服务器节点上,当节点发生故障异常退出或系统发生较大的负载倾斜时可以及时检测,并可通过任务迁移等手段迅速调整,并保证该过程尽可能平滑,对用户透明。

(2)录制服务器用来执行调度器分配的具体的录制任务,监听并接收会话的音视频组播网络上的 RTP 报文,经缓冲处理后,将 RTP 报文存储下来。为保证系统的实时性,文中在录制之后才对数据文件进行编解码并最终转换为标准流媒体格式。同时,录制服务

器定时向中心服务器报告自己的负载,当后者发送任务迁移的命令时,录制服务器进行任务迁移或接收迁移任务的准备。录制服务器模块关系如图 2 所示。

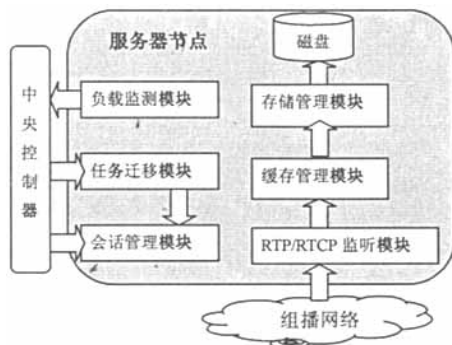


图 2 服务器节点模块图

(3)监听器向用户返回欲录制的会话的当前流列表,即有哪些与会者,以及其地址、描述等信息,供用户选择性地录制,当用户动态加入或退出会议时,监听器可以及时地通知客户端。监听器采用了订阅-发布模式,会话中流的信息通过解析 SDPS 以及 BYE 类型的 RTCP 的报文来获取。

(4)客户端为用户提供了会议控制及状态信息查看的接口。

### 1.2 工作流程

一个完整的会话录制工作及系统各模块交互过程如图 3 所示。

- 1)客户端向监听器订阅会话 S 的视频流列表;
- 2)监听器向客户端返回会话 S 可录制的音视频流列表;
- 3)客户端选择录制源并向中央控制器发送录制请求;
- 4)中央控制器根据服务器节点最新负载状态报告以及一定的负载评估算法,将任务分配到条件最合适的服务器节点;
- 5)该节点监听会话对应的组播地址,并将相应的 RTP 报文按照一定的存储策略存储;
- 6)当中央控制器监测到系统负载不再平衡时,则将重负载节点上的适当的任务动态迁移到另一个空闲节点上。

## 2 录制关键技术

### 2.1 缓冲管理

RTP 协议基于不可靠的 UDP 传输协议,因而从

相应的组播网络上接收的 RTP 报文可能会乱序;另外,随着并发流的增多,从网络上接收的数据可能来不及写入磁盘而造成丢包。综上两点,在 RTP 报文写入文件前应进行缓冲处理。

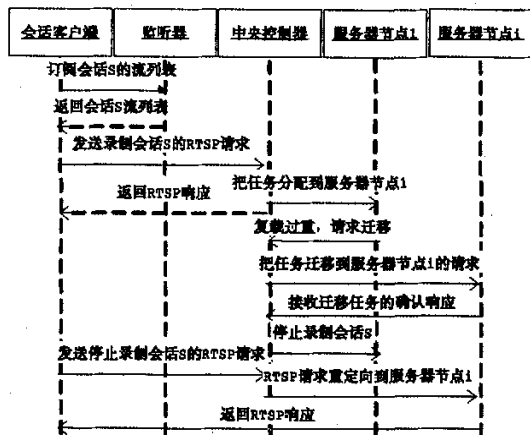


图3 工作流程图

文中对来自同一会话统一媒体源的报文进行缓冲排序,系统根据 RTP 报文头的序列号(SEQ)及时间戳(TimeStamp)字段进行排序调整。此外,在实验过程中发现,系统网络接口缓存区的大小也会影响系统吞吐量。由于视频中的 I 帧相比 P 帧、B 帧更大(一般地,I 帧可能封装在多个 RTP 报文中,而每个报文约为 1k 左右),当并发视频流增多或者发送端增加数据带宽时都容易引起网络上 I 帧报文激增。而系统缓存区默认大小为 16k,很容易造成缓存溢出,从而丢包。实验表明,随着并发数量的增加,系统丢包率明显上升,吞吐率下降。当系统缓存设置为 256k 时,已经能基本满足条件。

## 2.2 媒体存储策略与文件组织

如前文所述,录制任务为 I/O 集中性任务,I/O 是录制服务器系统性能的最大瓶颈,所以需要合理的数据组织方案来保证录制的实时性与连续性,避免由于 I/O 繁忙引起数据溢出或丢失。目前提出的很多方案<sup>[3]</sup>多是基于较为高端的硬件环境,如基于 SCSI 接口的 Raid 阵列等提出了条带式存储(Data Strip)数据间隔(Data Interleaving)等比较底层的数据组织优化方案或者依赖于专业的媒体文件系统,而文中则更针对低端,但更为广泛廉价的硬盘存储环境,使得中小型协同环境更容易部署,因而本系统的媒体存储策略更多关注数据组织策略以及文件写方式等方面的问题。

为提高 I/O 效率,根据流的唯一标识 SSRC 把同一会话的所有 RTP 流随机分成  $n$  簇,不同簇存放在同一个服务器节点的不同磁盘中,这样,I/O 访问并行化,减小了磁盘负载,并且控制简单统一。进一步地,

把同一个簇中的视频流报文连续存储在同一文件中,该法相比每路视频存为单独文件的思路,大大减少磁头移动次数,进而提高了系统并发数量,且随着会议规模的增大,该优势更加明显。另外,文中将一个会话的音频和视频的 RTP 流分开存储以便于编码时的音视频合成。

如上文所述,同一个簇中的 RTP 报文按照缓存排序后的顺序存储下来。为了离线编码时对不同的视频流间进行同步,以还原会议当时各个节点的全部场景,在每个 RTP 报文前扩充一个 RTP Dump 头。其中的时间偏移域(time offset)是最重要的一个字段,其定义为当前报文与该会话录制开始时的第一个报文到达本服务器节点的时间间隔,该域用于表征不同源间的同步关系,即时间偏移相同而源不同的媒体流被认为是该会话在同时刻不同节点的音视频内容。这样可根据时间偏移来进行。由于来自不同源节点的 RTP 报文的时间戳参考的机器时钟不同,所以文中以报文接收端的服务器节点时钟为统一参考标准。

在写文件方式上,本系统采用了更为高效的内存映射文件技术,因为相比普通的 I/O 文件读写,该法省去了文件缓冲过程,大大提高了 I/O 吞吐量,更加适合海量数据大文件的处理。另外,由于数据文件较大,容易出错;且使用内存映射文件技术在系统异常退出时也会带来大片无用数据,浪费存储。因而笔者通过一个恢复参考文件来进行简单地容错,在编码前可以先根据该文件进行错误检测和恢复。同时,系统还为每个会话建立一个位置索引文件,把由于分流、任务迁移,或者超过文件大小限制而新建的文件等分散在多处文件序列串起来,便于连续访问。

## 2.3 会话状态管理

多媒体协作以逻辑会话为单元,而会话是基于状态的,来自相同会话的先后请求(如开始或停止录制请求)有一定的逻辑连续性。另外,并发录制的多个会话之间要确保彼此独立,互不干扰。基于上述分析,在中心控制器和服务器节点都设计了会话管理模块。中央控制器负责把同一会话的所有请求定向到同一个录制服务器实例,并跟踪任务迁移等引起的会话分布状态变更。另外,中央控制器定义了一个 XML 文件(如图 4 所示)录制所有会话的永久信息,便于控制器对节点数据分布的了解,为其做出合理的调度策略提供依据。该文件包括会话名称、基本描述、录制时间、所有的媒体流标识,以及活动者信息等元素,具体定义如图 4。而录制服务器为每个会话启动一个独立线程,并建立一个哈希表来维护会话与流的映射关系。同时,录制服务器实时检测会话状态的变化,当与会者全部退出

时,系统自动结束对该会话的录制从而减少资源浪费。

```

<Server>
  <Description></Description>
  <Session>
    <Record Time>
      <Video>
        <stream 1>
          <ssrc></ssrc>
          <description></description>
          <path></path>
          ...
          <path></path>
        </stream 1>
      </Video>
      <Audio>
        <path></path>
        ...
        <path></path>
      </Audio>
    </Session>
    ...
  </Session>...</Session>
</Server>
...
<Server>...</Server>

```

图 4 会话录制描述文件定义

### 3 负载均衡问题

#### 3.1 动态负载均衡机制

为了最大程度利用集群里的多台服务器,在中心控制器上采用了动态负载均衡机制来保证录制负载分布基本均匀。文中提出的负载均衡策略主要考虑了两个决策因素:首先,负载指标应能最好地反映节点当前负载<sup>[4]</sup>。选择 CPU、内存资源、磁盘空间三种资源利用率<sup>[5,6]</sup>为负载因子,节点负载则为各因子的线性加权和,即  $load = W_{cpu} * E_{cpu} + W_{mem} * E_{mem} + W_{disk} * E_{disk}$ ,其中  $W_i$  为权重,  $E_i$  为资源利用率。各因子的权重根据录制任务对不同资源的需求的重要性设定,其为经验值。实验表明,本系统的最优权重组合为  $\{0.35, 0.2, 0.45\}$ ,则负载计算公式为:

$$load = 0.35 * E_{cpu} + 0.2 * E_{mem} + 0.45 * E_{disk} \quad (1)$$

该式反映了 CPU 利用率和磁盘利用率是影响系统负载的主要因素。另一方面,除了考虑负载之外,还要动态估计节点上的可用存储空间,以及其是否满足执行新任务的最小资源需求(在文中,最小资源定义为该会话录制至少 30 分钟所需的存储),以防止所分配的节点由于资源不足而引起任务在多个节点上的频繁迁移,带来系统抖动。

文中提出的负载均衡策略过程如下:

- 1) 通过负载公式(1) 计算各节点负载;
- 2) 从所有节点中选出  $K$  个负载最小的节点,实验表明,  $K$  一般取值为  $\lceil \log_2 n \rceil$ ,  $n$  为服务器节点数量;
- 3) 根据每个服务器节点当前录制任务的数量以及每个会话的进度,估计节点的可用存储空间,并根据

欲录制会话的数据带宽预测执行该会话录制任务的最小资源需求量  $R_{min}$ ;

4) 从  $K$  子集中选出存储资源大于  $R_{min}$  的节点,为防止位置群聚效应,本系统采用随机选择的方式;

5) 如果没有满足上述条件的节点,则放弃录制该会话并报告用户。

#### 3.2 录制任务的动态迁移

另外,考虑到在录制任务分配到节点后的执行过程中,仍可能因与会者动态加入或退出等引起节点负载动态变化,使系统不再平衡,如果系统负载过于倾斜,则需要把重负载节点的部分任务迁移到轻负载上。为防止不必要的迁移,根据实验设立负载阈值把节点分为几类,其中负载大于 0.7 时为重载,小于 0.35 时为轻载。迁移过程由重载节点发起。首先,重载节点根据其承载的所有会话任务的进度、数据带宽等影响节点负载的主要因素,选择一个用以迁移的目标会话任务,并主动向中心调度器请求转移该会话任务;接着,由中央调度器决策迁移目的地,策略与 3.1 节的负载分配机制类似,并通知该目标节点;当目标节点建立好相应的录制环境后,开始录制迁移会话,并通知中央控制器以及源重载节点;最终,源重载节点停止录制该会话。这样,录制任务就被重定向到了另一个负载较轻的节点上。由上述流程看出,整个迁移过程连续平滑,没有丢包,保证了对用户的透明。对于重叠录制的冗余数据可以在脱线编码时进行处理。

### 4 总结与展望

分析了多媒体实时协作平台中的录制系统面临数据量大、实时性高、并发性多等几大困难,进而研究并实现了一个基于 RTP 协议的录制集群系统。该系统面向个人桌面环境,成本低,更容易部署。就缓冲管理、数据存储、文件组织以及会话管理等录制中的关键问题进行了分析并给出了有效的解决方案;为了提高系统伸缩性,最大化利用率,提出了基于负载信息和存储预测的动态负载均衡机制以及任务迁移策略。目前,该系统已经配合 Admire 系统在很多实际环境中得到应用。

但也存在一些不足,如数据容错机制比较简单,对各录制服务器节点的位置分布没有进行很多考虑,另外在负载均衡策略的设计中忽略了通信延时的代价。

#### 参考文献:

- [1] Disz T, Judson I, Olson R, et al. The Argonne Voyager Multimedia Server[C]//6th IEEE International Symposium on

(下转第 170 页)

```
tion; // objectName 为性能对象名
}
```

web 服务器在系统中处于客户端,通过调用接口定义的方法获得采集数据:

```
GetInterface obj = (GetInterface) Naming.lookup
("rmi://192.168.88.6:1099/Server");
String message;
message = obj.get("CPU"); //获得 CPU 的采集数据
```

将所采集的性能数据解析,并按指定格式存储到数据库中,由于入库是每个采集类均有的操作,所以将其写成一个公共的 InputLibrary 类。每个采集类只需向其传送参数就能将数据入库。

## 2.5 Applet 技术

Applet 是一种特殊的 Java 程序,它通常通过支持 Java 的网页浏览器下载后执行。Applet 非常适用于浏览器上动态图形的呈现<sup>[2]</sup>。利用 Applet 技术生成基于 web 的网络拓扑图可以实现很好的显示效果。

## 2.6 网络拓扑生成

拓扑生成需要进行大量网络节点的计算,设计一个性能良好的算法是必需的。系统采用的算法如下:

- (1)生成所有的路由器节点;
- (2)根据某路由器节点生成其子网的所有服务器节点;
- (3)转第(2)步,直到生成完所有的服务器节点。

(上接第 162 页)

## 4 结 论

对 PIM-DM 协议进行了具体分析并修改、添加对批量密钥更新管理,并对密钥树进行批量密钥更新仿真,对实时和批量两种更新策略进行了比较,结果在提高密钥更新效率的基础上增强 PIM-DM 协议的安全性,增大了组规模的可扩展性,因而适用于大型的动态多播环境。

### 参考文献:

- [1] 许 勇,凌 龙,顾冠群.可靠可缩放安全多播密钥更新实现研究[J].计算机研究与发展,2004,41(6):934-939.

(上接第 166 页)

- High Performance Distributed Computing. [s. l.]: [s. n.], 1997.
- [2] Tian Jin, Sheng Xiangzhi, Lu Jian. Admire - A Prototype of Large Scale E-collaboration Platform, Grid and Cooperative Computing[C]//Second International Workshop. [s. l.]: [s. n.], 2003:335-343.
- [3] Gemmell D J. Multimedia Storage Servers: A Tutorial and Survey[J]. IEEE Computer, 1995, 28(5):40-49.

本算法的优点是:a.分级生成路由器和服务器节点方便整个网络拓扑的定位;b.生成所有子网中的服务器节点可以同时进行,提高图形生成效率。

## 3 结 语

所构建的系统采用三层结构,通过浏览器对广域网中服务器的性能进行实时监控,实现了拓扑生成和图形显示等功能。将系统在校园网上进行 7×24 不间断试运行,运行情况良好,具有较好的图形显示效果,这表明系统的设计是灵活和健壮的。

### 参考文献:

- [1] 姜劲松,吴礼发,张 萍.基于 WMI 的系统管理的设计与实现[J].计算机应用,2004,24(3):16-18.
- [2] Douglas F, Foster I. JAVA Tutorial[EB/OL]. 2003-06-12. <http://java.sun.com>.
- [3] Patrol for Microsoft Windows Servers Release Notes[EB/OL]. 2000-12-07. <http://www.bmc.com>.
- [4] Foster I, Kesselman C, Nick J M. In-Depth Guidance for Securing Windows 2000 Server[EB/OL]. 2002-03-10. <http://www.microsoft.com/windows2000/en/server/help/default.asp?url=/windows2000/en/server/help/windows-WMI-overview.htm?id=7>.
- [5] Windows 2000 Resource Kit [EB/OL]. 2003-04-26. <http://java.microsoft.com>.

- [2] Li X S, Yang Y R, Gouda M G, et al. Batch rekeying for secure group communications[C]//The 10th Int'l World Wide Web Conference. Hong Kong: [s. n.], 2001.
- [3] Adams A, Nicholas J, Siadak W. Protocol Independent Multicast-Dense Mode (PIM-DM): Protocol Specification (Revised)[S]. IETF, RFC3973, 2005.
- [4] Greis M. Tutorial for the Network Simulator ns[EB/OL]. 2000. <http://www.isi.edu/nsnam/ns/tutorial/index.html>.
- [5] Fall K, Varadhan K. The ns Manual (formerly ns Notes and Documentation) [EB/OL]. 2003. <http://www.isi.edu/nsnam/ns/ns-documentation.html>.
- [6] Altman E, Jimenez T. NS Simulator for beginners[J]. Lecture notes, 2003, 12:29-31.

- [4] Perez J M, Garcia F, Carrerero J. Data Allocation and Load Balancing for Heterogeneous Cluster Storage Systems[C]//Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid. Tokyo: [s. n.], 2003.
- [5] 蒋 江,张民选,廖湘科.基于多种资源的负载均衡算法的研究[J].电子学报,2002,30(8):1148-1152.
- [6] 鞠九滨,杨 鲲,徐高潮.使用资源利用率作为负载均衡系统的负载指标[J].软件学报,1996,17(4):238-243.